



Chapter 4

Introduction to the Cisco IOS and SDM

IMPLEMENT A SMALL ROUTED NETWORK

- ✓ Access and utilize the router CLI to set basic parameters
 - ✓ Connect, configure, and verify operation status of a device interface
 - ✓ Verify device configuration and network connectivity using ping, traceroute, telnet, SSH or other utilities
 - ✓ Implement password and physical security
 - ✓ Verify network status and router operation using basic utilities (including: ping, traceroute, telnet, SSH, arp, ipconfig), SHOW & DEBUG commands
-

IMPLEMENT AN IP ADDRESSING SCHEME AND IP SERVICES TO MEET NETWORK REQUIREMENTS FOR A SMALL BRANCH OFFICE

- ✓ Describe the need and role of addressing in a network
- ✓ Create and apply an addressing scheme to a network
- ✓ Describe and verify DNS operation
- ✓ Configure, verify and troubleshoot DHCP and DNS operation on a router (including: CLI/SDM)



The time has come to introduce you to the Cisco Internetwork Operating System (IOS). The IOS is what runs Cisco routers as well as Cisco's switches, and it's what allows you to configure the devices as well.

So that's what you're going to learn about in this chapter. I'm going to show you how to configure a Cisco IOS router using the Cisco IOS command-line interface (CLI). When you become proficient with this interface, you'll be able to configure hostnames, banners, passwords, and more, as well as troubleshoot using the Cisco IOS.

From there, we'll take a peek at Cisco's Security Device Manager (SDM) and you'll find out how to set up a secure HTTPS session to a router to provide the same types of configurations. The SDM really becomes a much more powerful tool in more advanced topics because it makes configuring access lists, virtual private networks (VPNs), and IP Security (IPSec) a snap, but first, you need to learn the basics of Cisco's IOS.

I'm also going to get you up to speed on the vital basics of router configurations and command verifications. Here's a list of the subjects we'll be covering in this chapter:

- Understanding and configuring the Cisco Internetwork Operating System (IOS)
- Connecting to a router
- Bringing up a router
- Logging into a router
- Understanding the router prompts
- Understanding the CLI prompts
- Performing editing and help features
- Gathering basic routing information
- Setting administrative functions
- Setting hostnames
- Setting banners
- Setting passwords
- Setting interface descriptions
- Performing interface configurations
- Viewing, saving, and erasing configurations
- Verifying routing configurations

And just as it was with preceding chapters, the fundamentals that you'll learn in this chapter are foundational building blocks that really need to be in place before you go on to the next chapters in the book.



For up-to-the-minute updates for this chapter, please see www.lammle.com and/or www.sybex.com/go/ccent.

The IOS User Interface

The *Cisco Internetwork Operating System (IOS)* is the kernel of Cisco routers and most switches. In case you didn't know, a kernel is the basic, indispensable part of an operating system that allocates resources and manages things such as low-level hardware interfaces and security.

In the following sections, I'll show you the Cisco IOS and how to configure a Cisco router using the command-line interface (CLI). We'll use Cisco's SDM toward the end of the chapter.



I'm going to save Cisco switch configurations for Chapter 7, "Layer 2 Switching."

Cisco Router IOS

The Cisco IOS is a proprietary kernel that provides routing, switching, internetworking, and telecommunications features. The first IOS was written by William Yeager in 1986, and it enabled networked applications. It runs on most Cisco routers as well as an ever-increasing number of Cisco Catalyst switches, like the Catalyst 2950/2960 and 3550/3560 series switches.

These are some important things that the Cisco router IOS software is responsible for:

- Carrying network protocols and functions
- Connecting high-speed traffic between devices
- Adding security to control access and stop unauthorized network use
- Providing scalability for ease of network growth and redundancy
- Supplying network reliability for connecting to network resources

You can access the Cisco IOS through the console port of a router, from a modem into the auxiliary (or Aux) port, or even through Telnet. Access to the IOS command line is called an *EXEC session*.

Connecting to a Cisco Router

You can connect to a Cisco router to configure it, verify its configuration, and check statistics. There are different ways to do this, but most often, the first place you would connect to is the console port. The *console port* is usually an RJ-45 (8-pin modular) connection located at the back of the router—by default, there may or may not be a password set. The new ISR routers use *cisco* as the username and *cisco* as the password by default.



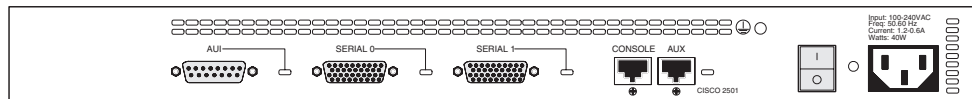
See Chapter 1, “Internetworking,” for an explanation of how to configure a PC to connect to a router console port.

You can also connect to a Cisco router through an *auxiliary port*—which is really the same thing as a console port, so it follows that you can use it as one. But an auxiliary port also allows you to configure modem commands so that a modem can be connected to the router. This is a cool feature—it lets you dial up a remote router and attach to the auxiliary port if the router is down and you need to configure it *out-of-band* (meaning out of the network).

The third way to connect to a Cisco router is in-band, through the program *Telnet*. (*In-band* means configuring the router through the network, the opposite of *out-of-band*.) Telnet is a terminal emulation program that acts as though it’s a dumb terminal. You can use Telnet to connect to any active interface on a router, such as an Ethernet or serial port.

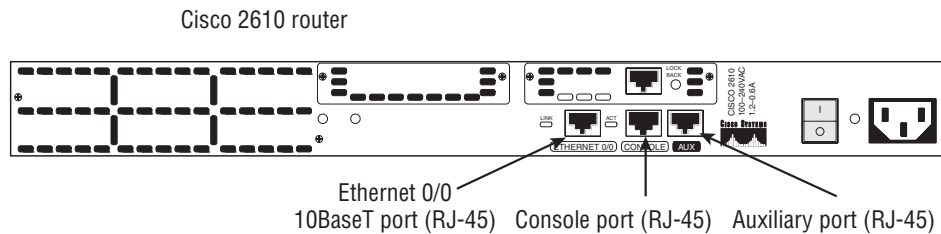
Figure 4.1 shows an illustration of a 2501 Cisco router. Pay close attention to all the different kinds of interfaces and connections.

FIGURE 4.1 A Cisco 2501 router



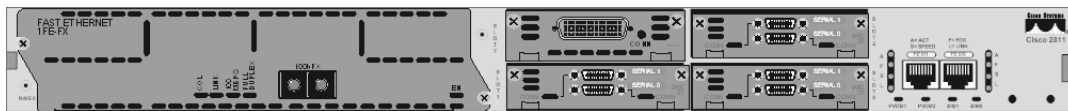
The 2501 router has two serial interfaces, which can be used for WAN connections, and one Attachment Unit Interface (AUI) connection for a 10Mbps Ethernet network connection. This router also has one console and one auxiliary connection via RJ-45 connectors. The 2501 router shown has two serial V.35 connections used for WAN connections like a T1, or Frame Relay. Plus, next to the console port could be another RJ45 port marked as BRI (Basic Rate Interface) that’s used with ISDN (Integrated Services Digital Network). And don’t stress—I’m going to cover T1, Frame Relay, ISDN, and more WAN protocols coming up in Chapter 10, “Introduction to Wide Area Networks.”

Figure 4.2 shows an illustration of a Cisco 2600 series modular router, which is a cut above routers populating the 2500 series because it has a faster processor and can handle many more interfaces. Both the 2500 and 2600 series routers are end of life (EOL), and you can only buy them used. However, many 2600 series routers are still found in production, so it’s important to understand them. Pay close attention to all the different kinds of interfaces and connections.

FIGURE 4.2 A Cisco 2600 router

The 2600 series router can have multiple serial interfaces, which can be used for connecting a T1 or Frame Relay using a serial V.35 WAN connection. Multiple Ethernet or FastEthernet ports can be used on the router, depending on the model. This router also has one console and one auxiliary connection via RJ-45 connectors.

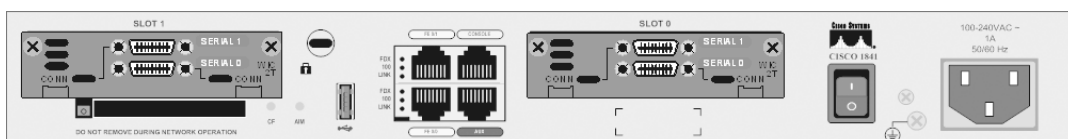
Another router I want to talk about is the 2800 series (shown in Figure 4.3). This router has replaced the 2600 series router series and is referred to as an Integrated Services Router (ISR). It gets its name because many of the services, like security, are built into it. It's a modular device like the 2600, but it's much faster and a lot more sleek—it's elegantly designed to support a broad new range of interface options.

FIGURE 4.3 A Cisco 2800 router

I mentioned that security is built in—the 2800 has the Security Device Manager (SDM) pre-installed. The SDM is a Web-based device-management tool for Cisco routers that can help you configure a router via a web console. I'll cover that later in this chapter. You need to keep in mind that for the most part, you get some serious bang for your buck with the 2800—unless you start adding a lot of interfaces to it. You've got to pony up for each one of those little beauties, and things can really start to add up—fast!

There are a couple of other series of routers that are less expensive than the 2800 series: the 1800 and 800 series. You may want to look into these routers if you're looking for a less-expensive alternative to the 2800 but still want to run the same 12.4 IOS and the latest SDM.

Figure 4.4 shows an 1841 router that holds most of the same interfaces as the 2800, but it's smaller and less expensive. The real reason you would opt for a 2800 instead of an 1800 series router comes down to the more advanced interfaces you can run on the 2800—things like the wireless controller and switching modules.

FIGURE 4.4 A Cisco 1841 router

As a heads up, I'm going to be using all new 2800, 1800, and 800 series routers throughout this book to demonstrate examples of router configurations. But understand that you can use the 2600 and even 2500 routers to practice routing principles.



You can find more information about all Cisco routers at www.cisco.com/en/US/products/hw/routers/index.html.

Bringing Up a Router

When you first bring up a Cisco router, it will run a power-on self-test (POST). If it passes, it will then look for and load the Cisco IOS from flash memory—if an IOS file is present. (Just in case you don't know, flash memory is electronically erasable programmable read-only memory—an EEPROM.) After that, the IOS loads and looks for a valid configuration—the startup-config—that's stored in nonvolatile RAM, or NVRAM.

The following messages appear when you first boot or reload a router (I am using my 2811 router):

```
System Bootstrap, Version 12.4(13r)T, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 2006 by cisco Systems, Inc.
Initializing memory for ECC
c2811 platform with 262144 Kbytes of main memory
Main memory is configured to 64 bit mode with ECC enabled
Upgrade ROMMON initialized
program load complete, entry point: 0x8000f000, size: 0xcb80
program load complete, entry point: 0x8000f000, size: 0xcb80
```

This is the first part of the router boot process output. It's information about the bootstrap program that first runs the POST. It then tells the router how to load, which by default is to find the IOS in flash memory. It also lists the amount of RAM in the router.

The next part shows us that the IOS is being decompressed into RAM:

```
program load complete, entry point: 0x8000f000, size: 0x14b45f8
Self decompressing the image :
#####
##### [OK]
```

The pound signs are telling us that the IOS is being loaded into RAM. After it is decompressed into RAM, the IOS is loaded and starts running the router, as shown below. Notice that the IOS version is stated as advanced security version 12.4(12):

[some output cut]

```
Cisco IOS Software, 2800 Software (C2800NM-ADVSECURITYK9-M), Version
12.4(12), RELEASE SOFTWARE (fc1)
```

```

Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by Cisco Systems, Inc.
Compiled Fri 17-Nov-06 12:02 by prod_rel_team
Image text-base: 0x40093160, data-base: 0x41AA0000

```

A sweet new feature of the new ISR routers is that the IOS name is no longer cryptic. The filename actually tells you what the IOS can do, as in Advanced Security. Once the IOS is loaded, the information learned from the POST will be displayed next, as you can see here:

```

[some output cut]
Cisco 2811 (revision 49.46) with 249856K/12288K bytes of memory.
Processor board ID FTX1049A1AB
2 FastEthernet interfaces
4 Serial(sync/async) interfaces
1 Virtual Private Network (VPN) Module
DRAM configuration is 64 bits wide with parity enabled.
239K bytes of non-volatile configuration memory.
62720K bytes of ATA CompactFlash (Read/Write)

```

There are two FastEthernet interfaces, four serial interfaces, plus a VPN module. The amount of RAM, NVRAM, and flash are also displayed. The above router output shows us that there's 256MB of RAM, 239K of NVRAM, and 64MB of flash.

When the IOS is loaded and up and running, a preconfiguration (called startup-config) will be copied from NVRAM into RAM. The copy of this file will be placed in RAM and called running-config.



My 1841 and 871W routers boot exactly the same as the 2811 router. The 1841 and 871W do show less memory and different interfaces, but other than that, they have the same bootup procedure and the same preconfigured startup-config file.

Bringing Up a Non-ISR Router (a 2600)

As you're about to see, the boot cycle is about the same for non-ISR routers as for the ISR routers. The following messages appear when you first boot or reload a 2600 router:

```

System Bootstrap, Version 11.3(2)XA4, RELEASE SOFTWARE (fc1)
Copyright (c) 1999 by cisco Systems, Inc.
TAC:Home:SW:IOS:Specials for info
C2600 platform with 65536 Kbytes of main memory

```

The next part shows us that the IOS is being decompressed into RAM:

```
program load complete, entry point:0x80008000, size:0x43b7fc
Self decompressing the image :
#####
#####
#####
#####
#####
#####
#####
## [OK]
```

So far, everything is pretty much the same. Notice below that the IOS version is stated as version 12.3(20):

```
Cisco Internetwork Operating System Software
IOS (tm) C2600 Software (C2600-IK903S3-M), Version 12.3(20), RELEASE
SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by cisco Systems, Inc.
Compiled Tue 08-Aug-06 20:50 by kesnyder
Image text-base: 0x80008098, data-base: 0x81A0E7A8
```

Just as with the 2800 series, once the IOS is loaded, the information learned from the POST will be displayed:

```
cisco 2610 (MPC860) processor (revision 0x202) with 61440K/4096K bytes
  of memory.
Processor board ID JAD03348593 (1529298102)
M860 processor: part number 0, mask 49
Bridging software.
X.25 software, Version 3.0.0.
1 Ethernet/IEEE 802.3 interface(s)
1 Serial network interface(s)
2 Serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
16384K bytes of processor board System flash (Read/Write)
```

Okay—finally what we see here is one Ethernet interface and three serial interfaces. The amount of RAM and flash is also displayed, and the above router output shows there are 64MB of RAM and 16MB of flash.

And as I mentioned, when the IOS is loaded and up and running, a valid configuration called the startup-config will be loaded from NVRAM. But here's where it differs from the default bootup of the ISR routers—if there isn't a configuration in NVRAM, the router will broadcast looking for a valid one on a TFTP host. (This can only happen if the router senses carrier detect, or CD, on any interface.) If the broadcast fails, it will then go into what is called *setup mode*—a step-by-step process to help you configure the router. So you need to remember that, if you plug any interface of your router into your network and then boot your router, you may have to wait a couple minutes while the router searches for the configuration.



You can have your ISR routers perform this boot cycle by erasing the startup-config and reloading the router. This will give you a clean router with no default configuration. I'll show you how to do this a little later in the chapter.

You can also enter setup mode at any time from the command line by typing the command `setup` from something called privileged mode, which I'll get to in a minute. Setup mode covers only some global commands and is generally just unhelpful. Here is an example:

```
Would you like to enter the initial configuration dialog? [yes/no]: y
```

```
At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[]'.
```

```
Basic management setup configures only enough connectivity
for management of the system, extended setup will ask you
to configure each interface on the system
```

```
Would you like to enter basic management setup? [yes/no]: y
Configuring global parameters:
```

```
Enter host name [Router]:Ctrl+C
Configuration aborted, no changes made.
```



You can exit setup mode at any time by pressing Ctrl+C.

I highly recommend going through setup mode once, then never again. You should always use the CLI or SDM.

Command-Line Interface (CLI)

I sometimes refer to the CLI as “Cash Line Interface” because, if you can create advanced configurations on Cisco routers and switches using the CLI, then you’ll get the cash!

To use the CLI, press Enter after the router finishes booting up. After you do that, the router will respond with messages that tell you all about the status of each and every one of its interfaces and then display a banner and ask you to log in. Here’s an example:

[some output cut]

```
*Feb 28 16:42:00.967: %VPN_HW-6-INFO_LOC: Crypto engine: onboard 0
  State changed to: Initialized
*Feb 28 16:42:00.971: %VPN_HW-6-INFO_LOC: Crypto engine: onboard 0
  State changed to: Enabled
*Feb 28 16:42:01.471: %LINK-3-UPDOWN: Interface FastEthernet0/0,
  changed state to up
*Feb 28 16:42:01.471: %LINK-3-UPDOWN: Interface FastEthernet0/1,
  changed state to up
*Feb 28 16:42:01.471: %LINK-3-UPDOWN: Interface Serial0/0/0, changed
  state to down
*Feb 28 16:42:01.471: %LINK-3-UPDOWN: Interface Serial0/0/1, changed
  state to down
*Feb 28 16:42:01.471: %LINK-3-UPDOWN: Interface Serial0/1/0, changed
  state to down
*Feb 28 16:42:01.471: %LINK-3-UPDOWN: Interface Serial0/2/0, changed
  state to down
```

[some output cut]

```
-----
Cisco Router and Security Device Manager (SDM) is installed on this
device. This feature requires the one-time use of the username
"cisco" with the password "cisco". The default username and password
have a privilege level of 15.
```

```
Please change these publicly known initial credentials using SDM or the
IOS CLI. Here are the Cisco IOS commands.
```

```
username <myuser> privilege 15 secret 0 <mypassword>
no username cisco
```

```
Replace <myuser> and <mypassword> with the username and password you
want to use.
```

```
For more information about SDM please follow the instructions in the
QUICK START GUIDE for your router or go to http://www.cisco.com/go/sdm
-----
```

User Access Verification

Username: **cisco**

Password: **cisco** [this won't show on your screen]

yourname#

From here, just log in using *cisco/cisco* as the username/password and you will be in privileged mode—something I'll talk about next.

Now, the reason there's a configuration already installed on the router is so that you can connect with the SDM via HTTPS without having to configure the router. Again, I'll be going through the preconfigured startup-config a bit later in this chapter.

Entering the CLI from a Non-ISR Router

After the interface status messages appear and you press Enter, the **Router>** prompt will appear. This is called *user exec mode* (user mode), and it's mostly used to view statistics, but it's also a stepping stone to logging in to privileged mode.

You can only view and change the configuration of a Cisco router in *privileged exec mode* (privileged mode), which you can enter with the **enable** command.

Here's how:

```
Router>enable
```

```
Router#
```

You now end up with a **Router#** prompt, which indicates that you're in *privileged mode*, where you can both view and change the router's configuration. You can go back from privileged mode into user mode by using the **disable** command, as seen here:

```
Router#disable
```

```
Router>
```

At this point, you can type **logout** from either mode to exit the console:

```
Router>logout
```

```
Router con0 is now available
```

```
Press RETURN to get started.
```



Remember, by deleting the default configuration of an ISR router and reloading, you'll get to these same prompts, and you won't be prompted for a username and password.

In the following sections, I am going to show you how to perform some basic administrative configurations.

Overview of Router Modes

To configure from a CLI, you can make global changes to the router by typing `configure terminal` (or `config t` for short), which puts you in global configuration mode and changes what's known as the running-config. A global command (a command run from global config) is set only once and affects the entire router.

You can type `config` from the privileged-mode prompt and then just press Enter to take the default of `terminal`, as seen here:

```
yourname#config
Configuring from terminal, memory, or network [terminal]? [press enter]
Enter configuration commands, one per line. End with CNTL/Z.
yourname(config)#
```

At this point, you make changes that affect the router as a whole (globally), hence the term *global configuration mode*. To change the running-config—the current configuration running in dynamic RAM (DRAM)—you use the `configure terminal` command, as I just demonstrated.

To change the startup-config—the configuration stored in NVRAM—you use the `configure memory` command (or `config mem` for short), which merges the startup-config file into the running-config file in RAM. If you want to change a router configuration stored on a TFTP host (which is covered in Chapter 5, “Managing a Cisco Internetwork”), you use the `configure network` command (or `config net` for short), which also merges the file with the running-config file in RAM.

The `configure terminal`, `configure memory`, and `configure network` commands are all used to configure information into RAM on a router; however, typically only the `configure terminal` command is used. It is possible, though, that the commands `config mem` and `config net` can be useful if you screw up your running-config file and don't want to reboot your router.

Here are some of the other options under the `configure` command:

```
yourname(config)#exit or press ctrl-z
yourname#config ?
  confirm          Confirm replacement of running-config with
a new config file
  memory           Configure from NV memory
  network          Configure from a TFTP network host
  overwrite-network Overwrite NV memory from TFTP network host
  replace          Replace the running-config with a new config file
  terminal         Configure from the terminal
  <cr>
```

As you can see, Cisco has added a few more commands in the 12.4 IOS. We'll go through these commands in Chapter 5.

CLI Prompts

It's really important that you understand the different prompts you can find when configuring a router. Knowing these well will help you navigate and recognize where you are at any time within configuration mode. In this section, I'm going to demonstrate the prompts that are used on a Cisco router and discuss the various terms used. (Always check your prompts before making any changes to a router's configuration!)

I'm not going into every different command prompt offered, because doing that would be reaching beyond the scope of this book. Instead, I'm going to describe all the different prompts you'll see throughout this chapter and the rest of the book. These command prompts really are the ones you'll use most in real life anyway; plus, they're the ones you'll need to know for the exam.



Don't freak! It's not important that you understand what each of these command prompts accomplishes yet because I'm going to completely fill you in on all of them really soon. So right now, just relax and focus on becoming familiar with the different prompts available and all will be well!

Interfaces

To make changes to an interface, you use the `interface` command from global configuration mode:

`yourname(config)#interface ?`

<code>Async</code>	<code>Async interface</code>
<code>BVI</code>	<code>Bridge-Group Virtual Interface</code>
<code>CDMA-Ix</code>	<code>CDMA Ix interface</code>
<code>CTunnel</code>	<code>CTunnel interface</code>
<code>Dialer</code>	<code>Dialer interface</code>
<code>FastEthernet</code>	<code>FastEthernet IEEE 802.3</code>
<code>Group-Async</code>	<code>Async Group interface</code>
<code>Lex</code>	<code>Lex interface</code>
<code>Loopback</code>	<code>Loopback interface</code>
<code>MFR</code>	<code>Multilink Frame Relay bundle interface</code>
<code>Multilink</code>	<code>Multilink-group interface</code>
<code>Null</code>	<code>Null interface</code>
<code>Port-channel</code>	<code>Ethernet Channel of interfaces</code>
<code>Serial</code>	<code>Serial</code>
<code>Tunnel</code>	<code>Tunnel interface</code>
<code>Vif</code>	<code>PGM Multicast Host interface</code>
<code>Virtual-PPP</code>	<code>Virtual PPP interface</code>
<code>Virtual-Template</code>	<code>Virtual Template interface</code>
<code>Virtual-TokenRing</code>	<code>Virtual TokenRing</code>

```

range                interface range command
yourname(config)#interface fastEthernet 0/0
yourname(config-if)#

```

Did you notice that the prompt changed to `yourname(config-if)#`? This tells you that you're in *interface configuration mode*. And wouldn't it be nice if the prompt also gave you an indication of what interface you were configuring? Well, at least for now we'll have to live without the prompt information, because it doesn't. One thing is for sure: You really have to pay attention when configuring a router!

Subinterfaces

Subinterfaces allow you to create logical interfaces within the router. The prompt then changes to `yourname(config-subif)#`:

```

yourname(config-if)#interface f0/0.1
yourname(config-subif)#

```



You can read more about subinterfaces in *CCNA Study Guide Exam 640-802, Sixth Edition* (Sybex, 2007).

Line Commands

To configure user-mode passwords, use the `line` command. The prompt then becomes `yourname(config-line)#`:

```

yourname#config t
Enter configuration commands, one per line. End with CNTL/Z.
yourname(config)#line ?
<0-337> First Line number
aux      Auxiliary line
console  Primary terminal line
tty      Terminal controller
vty      Virtual terminal
x/y      Slot/Port for Modems
x/y/z    Slot/Subslot/Port for Modems

```

The `line console 0` command is known as a major command (also called a *global command*), and any command typed from the `(config-line)` prompt is known as a subcommand.

Routing Protocol Configurations

To configure routing protocols such as RIP and EIGRP, you'll use the prompt `yourname(config-router#)`:

```
yourname#config t
Enter configuration commands, one per line.  End with CNTL/Z.
yourname(config)#router rip
yourname(config-router)#version 2
yourname(config-router)#
```

Defining Router Terms

Table 4.1 defines some of the terms we've used so far.

TABLE 4.1 Router Terms

Mode	Definition
User EXEC mode	Limited to basic monitoring commands
Privileged EXEC mode	Provides access to all other router commands
Global configuration mode	Commands that affect the entire system
Specific configuration modes	Commands that affect interfaces/processes only
Setup mode	Interactive configuration dialog

Editing and Help Features

You can use the Cisco advanced editing features to help you configure your router. If you type in a question mark (?) at any prompt, you'll be given a list of all the commands available from that prompt:

```
yourname#?
Exec commands:
  access-enable    Create a temporary Access-List entry
  access-profile   Apply user-profile to interface
  access-template  Create a temporary Access-List entry
  archive          manage archive files
  auto             Exec level Automation
  bfe              For manual emergency modes setting
```

```

calendar      Manage the hardware calendar
cd            Change current directory
clear        Reset functions
clock        Manage the system clock
cns          CNS agents
configure     Enter configuration mode
connect      Open a terminal connection
copy         Copy from one file to another
crypto       Encryption related commands.
ct-isdn      Run an ISDN component test command
debug        Debugging functions (see also 'undebug')
delete       Delete a file
dir          List files on a filesystem
disable      Turn off privileged commands
disconnect   Disconnect an existing network connection
--More--

```

Plus, at this point you can press the spacebar to get another page of information, or you can press Enter to go one command at a time. You can also press Q (or any other key, for that matter) to quit and return to the prompt.

Here's a shortcut: To find commands that start with a certain letter, use the letter and the question mark with no space between them:

```

yourname#c?
calendar  cd          clear   clock
cns       configure  connect  copy
crypto    ct-isdn

```

```
yourname#c
```

By typing `c?`, we received a response listing all the commands that start with `c`. Also notice that the `yourname#c` prompt reappears after the list of commands is displayed. This can be helpful when you have long commands and need the next possible command. It would be pretty lame if you had to retype the entire command every time you used a question mark!

To find the next command in a string, type the first command and then a question mark:

```

yourname#clock ?
  read-calendar  Read the hardware calendar into the clock
  set            Set the time and date
  update-calendar Update the hardware calendar from the clock
yourname#clock set ?
  hh:mm:ss      Current Time
yourname#clock set 11:15:11 ?
  <1-31>        Day of the month

```

```

MONTH    Month of the year
yourname#clock set 11:15:11 25 aug ?
<1993-2035> Year
yourname#clock set 11:15:11 25 aug 2007 ?
<cr>
yourname#clock set 11:15:11 25 aug 2007
*Aug 25 11:15:11.000: %SYS-6-CLOCKUPDATE: System clock has
been updated from 18:52:53 UTC Wed Feb 28 2007 to 11:15:11
UTC Sat Aug 25 2007, configured from console by cisco on console.

```

By typing the `clock ?` command, you'll get a list of the next possible parameters and what they do. Notice that you should just keep typing a command, a space, and then a question mark until `<cr>` (carriage return) is your only option.

If you're typing commands and receive

```

yourname#clock set 11:15:11
% Incomplete command.

```

you'll know that the command string isn't done yet. Just press the up arrow key to redisplay the last command entered, and then continue with the command by using your question mark.

And if you receive the error

```

yourname(config)#access-list 110 permit host 1.1.1.1
                        ^

```

```
% Invalid input detected at '^' marker.
```

you've entered a command incorrectly. See that little caret—the `^`? It's a very helpful tool that marks the exact point where you blew it and entered the command incorrectly. Here's another example of when you'll see the caret:

```

yourname#sh serial 0/0/0
                ^

```

```
% Invalid input detected at '^' marker.
```

This command looks right, but be careful! The problem is that the full command is `show interface serial 0/0/0`.

Now if you receive the error

```

yourname#sh ru
% Ambiguous command: "sh ru"

```

it means there are multiple commands that begin with the string you entered and it's not unique. Use the question mark to find the command you need:

```

yourname#sh ru?
rudpvl  running-config

```

As you can see, there are two commands that start with `show ru`.

Table 4.2 lists the enhanced editing commands available on a Cisco router.

TABLE 4.2 Enhanced Editing Commands

Command	Meaning
Ctrl+A	Moves your cursor to the beginning of the line
Ctrl+E	Moves your cursor to the end of the line
Esc+B	Moves back one word
Ctrl+B	Moves back one character
Ctrl+F	Moves forward one character
Esc+F	Moves forward one word
Ctrl+D	Deletes a single character
Backspace	Deletes a single character
Ctrl+R	Redisplays a line
Ctrl+U	Erases a line
Ctrl+W	Erases a word
Ctrl+Z	Ends configuration mode and returns to EXEC
Tab	Finishes typing a command for you

Another cool editing feature I want to show you is the automatic scrolling of long lines. In the following example, the command typed had reached the right margin and automatically moved 11 spaces to the left (the dollar sign [\$] indicates that the line has been scrolled to the left):

```
yourname#config t
Enter configuration commands, one per line. End with CNTL/Z.
yourname(config)#$110 permit host 171.10.10.10 0.0.0.0 eq 23
```

You can review the router command history with the commands shown in Table 4.3.

TABLE 4.3 Router Command History

Command	Meaning
Ctrl+P or up arrow	Shows last command entered
Ctrl+N or down arrow	Shows previous commands entered
show history	Shows last 10 commands entered by default
show terminal	Shows terminal configurations and history buffer size
terminal history size	Changes buffer size (max 256)

The following example demonstrates the `show history` command and how to change the history size, as well as how to verify it with the `show terminal` command. First, use the `show history` command to see the last 20 commands that were entered on the router:

```
yourname#show history
en
sh history
show terminal
sh cdp neig
sh ver
sh flash
sh int fa0
sh history
sh int s0/0
sh int s0/1
```

Now use the `show terminal` command to verify the terminal history size:

```
yourname#show terminal
Line 0, Location: "", Type: ""
[output cut]just codesnippet
Modem type is unknown.
Session limit is not set.
Time since activation: 00:21:41
Editing is enabled.
History is enabled, history size is 20.
DNS resolution in show commands is enabled
Full user help is disabled
Allowed input transports are none.
```

Allowed output transports are pad telnet rlogin lapb-ta mop v120 ssh.

Preferred transport is telnet.

No output characters are padded

No special data dispatching characters

The `terminal history size` command, used from privileged mode, can change the size of the history buffer:

```
yourname#terminal history size ?
```

```
<0-256> Size of history buffer
```

```
yourname#terminal history size 25
```

You verify the change with the `show terminal` command:

```
yourname#show terminal
```

```
Line 0, Location: "", Type: ""
```

```
[output cut]
```

```
Editing is enabled.
```

```
History is enabled, history size is 25.
```

```
Full user help is disabled
```

```
Allowed transports are lat pad v120 telnet mop rlogin
```

```
  nasi. Preferred is lat.
```

```
No output characters are padded
```

```
No special data dispatching characters
```

```
Group codes: 0
```



Real World Scenario

When Do You Use the Cisco Editing Features?

A couple of editing features are used quite often and some not so much, if at all. Understand that Cisco didn't make these up; these are just old Unix commands. However, `Ctrl+A` is really helpful to negate a command.

For example, if you were to put in a long command and then decide you didn't want to use that command in your configuration after all, or if it didn't work, then you could just press your up arrow key to show the last command entered, press `Ctrl+A`, type **no** and then a space, press Enter—and poof! The command is negated. This doesn't work on every command, but it works on a lot of them.

Gathering Basic Routing Information

The `show version` command will provide basic configuration for the system hardware as well as the software version and the boot images. Here's an example:

```
yourname#show version
Cisco IOS Software, 2800 Software (C2800NM-ADVSECURITYK9-M), Version
 12.4(12), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by Cisco Systems, Inc.
Compiled Fri 17-Nov-06 12:02 by prod_rel_team
```

The preceding section of output describes the Cisco IOS running on the router. The following section describes the read-only memory (ROM) used to boot the router and hold the POST:

```
ROM: System Bootstrap, Version 12.4(13r)T, RELEASE SOFTWARE (fc1)
```

The next section shows how long the router has been running, how it was restarted (if you see a `system restarted by bus error`, that is a very bad thing), where the Cisco IOS was loaded from, and the IOS name. Flash is the default:

```
yourname uptime is 2 hours, 30 minutes
System returned to ROM by power-on
System restarted at 09:04:07 UTC Sat Aug 25 2007
System image file is "flash:c2800nm-advsecurityk9-mz.124-12.bin"
```

This next section displays the processor, the amount of DRAM and flash memory, and the interfaces the POST found on the router:

```
[some output cut]
Cisco 2811 (revision 53.50) with 249856K/12288K bytes of memory.
Processor board ID FTX1049A1AB
2 FastEthernet interfaces
4 Serial(sync/async) interfaces
1 Virtual Private Network (VPN) Module
DRAM configuration is 64 bits wide with parity enabled.
239K bytes of non-volatile configuration memory.
62720K bytes of ATA CompactFlash (Read/Write)
Configuration register is 0x2102
```

The configuration register value is listed last—it's something I'll cover in Chapter 5.

In addition, the `show interfaces` and `show ip interface brief` commands are very useful in verifying and troubleshooting a router as well as network issues. These commands are covered later in this chapter. Don't miss it!

Router and Switch Administrative Configurations

Even though this section isn't critical to making a router or switch work on a network, it's still really important; in it, I'm going to lead you through configuring commands that will help you administer your network.

The administrative functions that you can configure on a router and switch are as follows:

- Hostnames
- Banners
- Passwords
- Interface descriptions

Remember, none of these will make your routers or switches work better or faster, but trust me, your life will be a whole lot better if you just take the time to set these configurations on each of your network devices. That's because doing this makes troubleshooting and maintaining your network sooooo much easier—seriously! In this next section, I'll be demonstrating commands on a Cisco router, but these commands are exactly the same on a Cisco switch.

Hostnames

You can set the identity of the router with the `hostname` command. This is only locally significant, which means that it has no bearing on how the router performs name lookups or how the router works on the internetwork. However, I'll use the `hostname` in Chapter 10 for authentication purposes when I discuss PPP (Point-to-Point Protocol).

Here's an example:

```
yourname#config t
Enter configuration commands, one per line. End with
CNTL/Z.
yourname(config)#hostname Todd
Todd(config)#hostname Atlanta
Atlanta(config)#hostname Todd
Todd(config)#
```

Even though it's pretty tempting to configure the `hostname` after your own name, it's definitely a better idea to name the router something pertinent to the location. This is because giving it a `hostname` that's somehow relevant to where the device actually lives will make finding it a whole lot easier. And it also helps you confirm that you are, indeed, configuring the right device. For this chapter, we'll leave it at Todd for now.

Banners

A *banner* is more than just a little cool—one very good reason for having a banner is to give any and all who dare attempt to telnet or dial into your internetwork a little security notice. And you can create a banner to give anyone who shows up on the router exactly the information you want them to have.

Make sure you're familiar with these four available banner types: exec process creation banner, incoming terminal line banner, login banner, and message of the day banner (all illustrated in the following code):

Todd(config)#**banner ?**

```
LINE          c banner-text c, where 'c' is a delimiting character
exec          Set EXEC process creation banner
incoming      Set incoming terminal line banner
login         Set login banner
motd          Set Message of the Day banner
prompt-timeout Set Message for login authentication timeout
slip-ppp      Set Message for SLIP/PPP
```

Message of the day (MOTD) is the most extensively used banner. It gives a message to every person dialing into or connecting to the router via Telnet or an auxiliary port, or even through a console port as seen here:

Todd(config)#**banner motd ?**

```
LINE c banner-text c, where 'c' is a delimiting character
```

Todd(config)#**banner motd #**

Enter TEXT message. End with the character '#'.
\$ Acme.com network, then you must disconnect immediately.

#

Todd(config)#**^Z**

Todd#

```
00:25:12: %SYS-5-CONFIG_I: Configured from console by
console
```

Todd#**exit**

Router con0 is now available

Press RETURN to get started.

```
If you are not authorized to be in Acme.com network, then you must
disconnect immediately.
```

Todd#

The preceding MOTD banner essentially tells anyone connecting to the router to get lost if they're not on the guest list! The part to understand is the delimiting character—the thing that's used to tell the router when the message is done. You can use any character you want for it, but (I hope this is obvious) you can't use the delimiting character in the message itself. Also, once the message is complete, press Enter, then the delimiting character, and then Enter again. It'll still work if you don't do that, but if you have more than one banner, they'll be combined as one message and put on a single line.

For example, you can set a banner on one line as shown:

```
Todd(config)#banner motd x Unauthorized access prohibited! x
```

This example will work just fine, but if you add another MOTD banner message, they would end up on a single line.

Here are some details of the other banners I mentioned:

Exec banner You can configure a line-activation (exec) banner to be displayed when an EXEC process (such as a line activation or incoming connection to a VTY line) is created. By simply starting a user exec session through a console port, you'll activate the exec banner.

Incoming banner You can configure a banner to be displayed on terminals connected to reverse Telnet lines. This banner is useful for providing instructions to users who use reverse Telnet.

Login banner You can configure a login banner to be displayed on all connected terminals. This banner is displayed after the MOTD banner but before the login prompts. The login banner can't be disabled on a per-line basis, so to globally disable it, you've got to delete it with the `no banner login` command.

Here is an example of a login banner:

```
!
banner login ^C
-----
Cisco Router and Security Device Manager (SDM) is installed on this device.
This feature requires the one-time use of the username "cisco"
with the password "cisco". The default username and password have a privilege
level of 15.
Please change these publicly known initial credentials using SDM or the IOS
CLI.
Here are the Cisco IOS commands.
username <myuser> privilege 15 secret 0 <mypassword>
no username cisco
Replace <myuser> and <mypassword> with the username and password you want to
use.
For more information about SDM please follow the instructions in the QUICK
START
```

```
GUIDE for your router or go to http://www.cisco.com/go/sdm
```

```
-----
```

```
^C
```

```
!
```

The above login banner should look pretty familiar—it’s the banner that Cisco has in its default configuration for its ISR routers. Again, this banner is displayed before the login prompts but *after* the MOTD banner.

Setting Passwords

Five passwords are used to secure your Cisco routers: console, auxiliary, telnet (VTY), enable password, and enable secret. The enable secret and enable password are used to set the password that’s used to secure privileged mode. This will prompt a user for a password when the `enable` command is used. The other three are used to configure a password when user mode is accessed through the console port, through the auxiliary port, or via Telnet.

Let’s take a look at each of these now.

Enable Passwords

You set the enable passwords from global configuration mode like this:

```
Todd(config)#enable ?
```

```
last-resort Define enable action if no TACACS servers
              respond
```

```
password Assign the privileged level password
```

```
secret Assign the privileged level secret
```

```
use-tacacs Use TACACS to check enable passwords
```

The following points describe the enable password parameters:

last-resort Allows you to still enter the router if you set up authentication through a TACACS (Terminal Access Controller Access System) server and it’s not available. But it isn’t used if the TACACS server is working.

password Sets the enable password on older, pre-10.3 systems, and isn’t ever used if an enable secret is set.

secret This is the newer, encrypted password that overrides the enable password if it’s set.

use-tacacs This tells the router to authenticate through a TACACS server. It’s convenient if you have anywhere from a dozen to multitudes of routers because, well, would you like to face the fun task of changing the password on all those routers? If you’re sane, no, you wouldn’t. So instead, just go through the TACACS server and you only have to change the password once—yeah!

Here's an example of setting the enable passwords:

```
Todd(config)#enable secret todd
```

```
Todd(config)#enable password todd
```

The enable password you have chosen is the same as your enable secret. This is not recommended. Re-enter the enable password.

If you try to set the enable secret and enable passwords the same, the router will give you a nice, polite warning to change the second password. If you don't have older legacy routers, don't even bother to use the enable password.

User-mode passwords are assigned by using the `line` command:

```
Todd(config)#line ?
```

```
<0-337> First Line number
aux      Auxiliary line
console  Primary terminal line
tty      Terminal controller
vtty     Virtual terminal
x/y      Slot/Port for Modems
x/y/z    Slot/Subslot/Port for Modems
```

Here are the lines to be concerned with:

aux Sets the user-mode password for the auxiliary port. It's usually used for attaching a modem to the router, but it can be used as a console as well.

console Sets a console user-mode password.

vtty Sets a Telnet password on the router. If this password isn't set, then Telnet can't be used by default.

To configure the user-mode passwords, you configure the line you want and use either the `login` or `no login` command to tell the router to prompt for authentication. The next sections will provide a line-by-line example of the configuration of each line.

Auxiliary Password

To configure the auxiliary password, go into global configuration mode and type `line aux ?`. You can see here that you only get a choice of 0-0 (that's because there's only one port):

```
Todd#config t
```

Enter configuration commands, one per line. End with CNTL/Z.

```
Todd(config)#line aux ?
```

```
<0-0> First Line number
```

```
Todd(config)#line aux 0
```

```
Todd(config-line)#login
```

```
% Login disabled on line 1, until 'password' is set
Todd(config-line)#password aux
Todd(config-line)#login
```

It's important to remember the `login` command or the auxiliary port won't prompt for authentication.

Notice that Cisco has begun this process of not letting you set the `login` command before a password is set on a line because, if you set the `login` command under a line and then don't set a password, the line won't be usable. And it will prompt for a password that doesn't exist. So this is a good thing—a feature, not a hassle!



Definitely remember that although Cisco has this new “password feature” on its routers starting in its newer IOS (12.2 and above), it's not in all of its IOSes.

Console Password

To set the console password, use the `line console 0` command. But look at what happened when I tried to type `line console 0 ?` from the `(config-line)#` prompt—I received an error. You can still type `line console 0` and it will accept it, but the help screens just don't work from that prompt. Type `exit` to get back one level and you'll find that your help screens now work. This is a “feature.” Really.

Here's the example:

```
Todd(config-line)#line console ?
% Unrecognized command
Todd(config-line)#exit
Todd(config)#line console ?
<0-0> First Line number
Todd(config-line)#password console
Todd(config-line)#login
```

Since there's only one console port, I can only choose line console 0. You can set all your line passwords to the same password, but for security reasons, I'd recommend that you make them different.

There are a few other important commands to know for the console port.

For one, the `exec-timeout 0 0` command sets the time-out for the console EXEC session to zero, which basically means to never time out. The default time-out is 10 minutes. (If you're feeling mischievous, try this on people at work: Set it to 0 1. That will make the console time out in 1 second! And to fix it, you have to continually press the down arrow key while changing the time-out time with your free hand!)

`logging synchronous` is a very cool command, and it should be a default command, but it's not. It stops annoying console messages from popping up and disrupting the input you're trying to type. The messages still pop up, but you are returned to your router prompt without your input interrupted. This makes your input messages oh-so-much easier to read.

Here's an example of how to configure both commands:

```
Todd(config-line)#line con 0
Todd(config-line)#exec-timeout ?
    <0-35791> Timeout in minutes
Todd(config-line)#exec-timeout 0 ?
    <0-2147483> Timeout in seconds
    <cr>
Todd(config-line)#exec-timeout 0 0
Todd(config-line)#logging synchronous
```



You can set the console to go from never timing out (0 0) to timing out in 35,791 minutes and 2,147,483 seconds. The default is 10 minutes.

Telnet Password

To set the user-mode password for Telnet access into the router, use the `line vty` command. Routers that aren't running the Enterprise edition of the Cisco IOS default to five VTY lines, 0 through 4. But if you have the Enterprise edition, you'll have significantly more. The best way to find out how many lines you have is to use that question mark:

```
Todd(config-line)#line vty 0 ?
% Unrecognized command
Todd(config-line)#exit
Todd(config)#line vty 0 ?
    <1-1180> Last Line number
    <cr>
Todd(config)#line vty 0 1180
Todd(config-line)#password telnet
Todd(config-line)#login
```

Remember, you cannot get help from your `(config-line)#` prompt. You must go back to privileged mode in order to use the question mark (?).



You may or may not have to set the `login` command before the password on the VTY lines—it depends on the IOS version. The result is the same either way.

So what will happen if you try to telnet into a router that doesn't have a VTY password set? You'll receive an error stating that the connection is refused because, well, the password isn't set. So, if you telnet into a router and receive the message

```
Todd#telnet SFRouter
```

```
Trying SFRouter (10.0.0.1)...Open
```

```
Password required, but none set
```

```
[Connection to SFRouter closed by foreign host]
```

```
Todd#
```

then the remote router (SFRouter in this example) does not have the VTY (Telnet) password set. But you can get around this and tell the router to allow Telnet connections without a password by using the `no login` command:

```
SFRouter(config-line)#line vty 0 4
```

```
SFRouter(config-line)#no login
```



I do not recommend using the `no login` command to allow Telnet connections without a password unless you are in a testing or classroom environment! In a production network, you should always set your VTY password.

After your routers are configured with an IP address, you can use the Telnet program to configure and check your routers instead of having to use a console cable. You can use the Telnet program by typing **telnet** from any command prompt (DOS or Cisco). Anything Telnet is covered more thoroughly in Chapter 5.

Setting Up Secure Shell (SSH)

Instead of Telnet, you can use Secure Shell, which creates a more secure session than the Telnet application that uses an unencrypted data stream. Secure Shell (SSH) uses encrypted keys to send data so that your username and password are not sent in the clear.



I recommend using SSH instead of Telnet whenever possible.

Here are the steps for setting up SSH:

1. Set your hostname:

```
Router(config)#hostname Todd
```

2. Set the domain name (both the hostname and domain name are required for the encryption keys to be generated):

```
Todd(config)#ip domain-name Lammle.com
```

3. Create a username and password on the router for login:

```
Todd(config)#username todd password lammle
```

4. Generate the encryption keys for securing the session:

```
Todd(config)#crypto key generate rsa general-keys modulus ?
<360-2048> size of the key modulus [360-2048]
Todd(config)#crypto key generate rsa general-keys modulus 1024
The name for the keys will be: Todd.Lammle.com
% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
*June 24 19:25:30.035: %SSH-5-ENABLED: SSH 1.99 has been enabled
```

5. Optionally set the max idle timer for a SSH session:

```
Todd(config)#ip ssh time-out ?
<1-120> SSH time-out interval (secs)
Todd(config)#ip ssh time-out 60
```

6. Optionally set the max failed attempts for an SSH connection:

```
Todd(config)#ip ssh authentication-retries ?
<0-5> Number of authentication retries
Todd(config)#ip ssh authentication-retries 2
```

7. Connect to the VTY lines of the router:

```
Todd(config)#line vty 0 1180
```

8. Configure the lines for local authentication:

```
Todd(config-line)#login local
```

9. Last, configure SSH as the access protocol:

```
Todd(config-line)#transport input ssh
```

If you want to use either SSH or Telnet on the device, then use the keyword `telnet` at the end of the command string:

```
Todd(config-line)#transport input ssh telnet
```

Here is a summary of the minimum SSH commands (it would be wise to remember this):

```
Router#config t
Router(config)#hostname Todd
```

```
Todd(config)#ip domain-name Lammle.com
Todd(config)#username todd password todd
Todd(config)#crypto key generate rsa general-keys modulus 1024
Todd(config)#line vty 0 4
Todd(config-line)#login local
Todd(config-line)#transport input SSH
```

Verifying the SSH service on your router

If you configure the basic SSH commands on your router, the router will now act as a SSH server and any host that can run a SSH client can connect. It's relatively simple.

From the router CLI, use the command `ssh -l username IP_address`. The *username* is the username created on the Todd router. You will then be prompted for a password, which is the password configured with the *username* command. Here is an example of using a SSH client on a Cisco router to connect to the Todd router that I configured above:

```
Sally#ssh 192.0.2.157
% No user specified nor available for SSH client
Sally#ssh -l todd 192.0.2.157
Trying 192.0.2.157...Open
```

Password:

Todd>

Notice that without the username specified that login is not allowed. Also, understand that the hostname of the router is irrelevant to the username set on the router.

Let's take a look at encrypting our passwords.

Encrypting Your Passwords

Because only the enable secret password is encrypted by default, you'll need to manually configure the user-mode and enable passwords for encryption.

Notice that you can see all the passwords except the enable secret when performing a `show running-config` on a router:

```
Todd#sh running-config
Building configuration...
[output cut]
!
enable secret 5 $1$2R.r$DcRaVo0yBnUJBf7dbG9XE0
enable password todd
!
```

[output cut]

```
!
line con 0
  exec-timeout 0 0
  password console
  logging synchronous
  login
line aux 0
  password aux
  login
line vty 0 4
  access-class 23 in
  privilege level 15
  password telnet
  login
  transport input telnet ssh
line vty 5 15
  access-class 23 in
  privilege level 15
  password telnet
  login
  transport input telnet ssh
line vty 16 1180
  password telnet
  login
!
end
```

To manually encrypt your passwords, use the `service password-encryption` command. Here's an example of how to do it:

```
Todd#config t
Enter configuration commands, one per line. End with CNTL/Z.
Todd(config)#service password-encryption
Todd(config)#exit
Todd#sh run
Building configuration...
[output cut]
!
enable secret 5 $1$2R.r$DcRaVo0yBnUJBf7dbG9XE0
enable password 7 131118160F
!
```

```
[output cut]
!
line con 0
  exec-timeout 0 0
  password 7 0605002F5F41051C
  logging synchronous
  login
line aux 0
  password 7 03054E13
  login
line vty 0 4
  access-class 23 in
  privilege level 15
  password 7 01070308550E12
  login
  transport input telnet ssh
line vty 5 15
  access-class 23 in
  privilege level 15
  password 7 01070308550E12
  login
  transport input telnet ssh
line vty 16 1180
  password 7 120D001B1C0E18
  login
!
end
```

```
Todd#config t
Todd(config)#no service password-encryption
Todd(config)#^Z
Todd#
```

There you have it! The passwords will now be encrypted. You just encrypt the passwords, perform a `show run`, and then turn off the command. You can see that the enable password and the line passwords are all encrypted.

I want to talk about encrypting passwords a bit more. As I said, if you set your passwords and then turn on the `service password-encryption` command, you have to perform a `show running-config` before you turn off the encryption service or your passwords won't be encrypted. You don't have to turn off the encryption service at all; you'd only do that if your router is running low on processes. And if you turn on the service before you set your passwords, then you don't even have to view them to get them encrypted.

But before I get into showing you all about setting descriptions on your routers, let's go through a quick summary of how you might set and encrypt your passwords.

1. Encrypt all clear text passwords: **service password-encryption**
2. Protect access to the user mode prompt: **line console 0; password friendOnly**
3. Set privileged mode encrypted password: **enable secret noWayIn**
4. Set password to allow Telnet connections: **line vty 0 4; password 2Hard**
5. Set privileged mode clear text password: **enable password dontuwish**
6. Set the VTY lines to only accept Secure Shell: **line vty 0 4; transport input ssh**



I am not recommending step 5, just showing you the Cisco objectives, and if you configure step 3, step 5 becomes irrelevant.

Descriptions

Setting descriptions on an interface is helpful to the administrator and, as with the hostname, only locally significant. The `description` command is a helpful one because you can, for instance, use it to keep track of circuit numbers.

Here's an example:

```
Todd#config t
Todd(config)#int s0/0/0
Todd(config-if)#description Wan to SF circuit number 6fdda12345678
Todd(config-if)#int fa0/0
Todd(config-if)#description Sales VLAN
Todd(config-if)#^Z
Todd#
```

You can view the description of an interface with either the `show running-config` command or the `show interface` command:

```
Todd#sh run
[output cut]
!
interface FastEthernet0/0
  description Sales VLAN
  ip address 10.10.10.1 255.255.255.248
  duplex auto
  speed auto
!
```

```

interface Serial0/0/0
description Wan to SF circuit number 6fdda 12345678
no ip address
shutdown
!
[output cut]
Todd#sh int f0/0
FastEthernet0/0 is up, line protocol is down
  Hardware is MV96340 Ethernet, address is 001a.2f55.c9e8 (bia 001a.2f55.c9e8)
  Description: Sales VLAN
[output cut]

Todd#sh int s0/0/0
Serial0/0/0 is administratively down, line protocol is down
  Hardware is GT96K Serial
  Description: Wan to SF circuit number 6fdda12345678

```



Real World Scenario

description: **A Helpful Command**

Bob, a senior network administrator at Acme Corporation in San Francisco, has over 50 WAN links to various branches throughout the U.S. and Canada. Whenever an interface goes down, Bob spends a lot of time trying to figure out the circuit number as well as the phone number of the provider of the WAN link.

The interface description command would be very helpful to Bob because he can use this command on his LAN links to discern exactly where every router interface is connected. And Bob would benefit tremendously by adding circuit numbers to each and every WAN interface, along with the phone number of the responsible provider.

So by spending the few hours it would take to add this information to each and every router interface, Bob can save a huge amount of precious time when his WAN links go down—and you know they will!

Doing the *do* Command

Beginning with IOS version 12.3, Cisco has finally added a command to the IOS that allows you to view the configuration and statistics from within configuration mode. (In the examples I gave you in the previous section, all **show** commands were run from privileged mode.)

In fact, with a pre-12.3 router, you'd get the following error if you tried to view the configuration from global-config:

```
Router(config)#sh run
                ^
% Invalid input detected at '^' marker.
```

Compare that to the output I get from entering that same command on my router that's running the 12.4 IOS:

Enter configuration commands, one per line. End with CNTL/Z.

```
Todd(config)#do show run
Building configuration...
```

```
Current configuration : 3276 bytes
```

```
!
```

```
[output cut]
```

```
Todd(config)#do sh int f0/0
```

```
FastEthernet0/0 is up, line protocol is down
```

```
Hardware is MV96340 Ethernet, address is 001a.2f55.c9e8 (bia
001a.2f55.c9e8)
```

```
Description: Sales VLAN
```

```
[output cut]
```

So basically, you can pretty much run any command from any configuration prompt now—cool, huh? Going back to the example of encrypting our passwords, the **do** command would definitely have gotten the party started sooner—so, my friends, this is a very, very good thing indeed!

Router Interfaces

Interface configuration is one of the most important router configurations because without interfaces, a router is pretty much a completely useless object. Plus, interface configurations must be totally precise to enable communication with other devices. Network layer addresses, media type, bandwidth, administrator commands are all used to configure an interface.

Different routers use different methods to choose the interfaces used on them. For instance, the following command shows a Cisco 2522 router with 10 serial interfaces, labeled 0 through 9:

```
Router(config)#int serial ?
<0-9> Serial interface number
```

Now it's time to choose the interface you want to configure. Once you do that, you will be in interface configuration for that specific interface. The following command would be used to choose serial port 5, for example:

```
Router(config)#int serial 5
Router(config)-if)#
```

The 2522 router has one Ethernet 10BaseT port, and typing **interface ethernet 0** can configure that interface, as seen here:

```
Router(config)#int ethernet ?
<0-0> Ethernet interface number
Router(config)#int ethernet 0
Router(config-if)#
```

As I showed you above, the 2500 router is a fixed-configuration router. This means that when you buy that model, you're stuck with that physical configuration—a huge reason why I don't use them much. I certainly never would use them in a production setting anymore.

To configure an interface, we always used the *interface type number* sequence, but with the 2600 and 2800 series routers (any ISR router for that matter), there's a physical slot in the router, with a port number on the module plugged into that slot. So on a modular router, the configuration would be *interface type slot/port*, as seen here:

```
Router(config)#int fastethernet ?
<0-1> FastEthernet interface number
Router(config)#int fastethernet 0
% Incomplete command.
Router(config)#int fastethernet 0?
/
Router(config)#int fastethernet 0/?
<0-1> FastEthernet interface number
```

Make note of the fact that you can't just type **int fastethernet 0**. You must type the full command: **type slot/port**, or **int fastethernet 0/0** (or **int fa 0/0**).

For the ISR series, it's basically the same, only you get even more options. For example, the built-in FastEthernet interfaces work with the same configuration we used with the 2600 series:

```
Todd(config)#int fastEthernet 0/?
<0-1> FastEthernet interface number
Todd(config)#int fastEthernet 0/0
Todd(config-if)#
```

But the rest of the modules are different—they use three numbers instead of two. The first 0 is the router itself, and then you choose the slot, and then the port. Here's an example of a serial interface on my 2811:

```
Todd(config)#interface serial ?
    <0-2> Serial interface number
Todd(config)#interface serial 0/0/?
    <0-1> Serial interface number
Todd(config)#interface serial 0/0/0
Todd(config-if)#
```

This can look a little dicey, I know, but I promise it's really not that hard! It helps to remember that you should always view a running-config output first so you know what interfaces you have to deal with. Here's my 2801 output:

```
Todd(config-if)#do show run
Building configuration...
[output cut]
!
interface FastEthernet0/0
    no ip address
    shutdown
    duplex auto
    speed auto
!
interface FastEthernet0/1
    no ip address
    shutdown
    duplex auto
    speed auto
!
interface Serial0/0/0
    no ip address
    shutdown
    no fair-queue
!
interface Serial0/0/1
    no ip address
    shutdown
!
interface Serial0/1/0
    no ip address
```

```

shutdown
!
interface Serial0/2/0
no ip address
shutdown
clock rate 2000000
!
[output cut]

```

For the sake of brevity, I didn't include my complete running-config, but I've displayed all you need. You can see the two built-in FastEthernet interfaces, the two serial interfaces in slot 0 (0/0/0 and 0/0/1), the serial interface in slot 1 (0/1/0), and the serial interface in slot 2 (0/2/0). Once you see the interfaces like this, it makes it a lot easier for you to understand how the modules are inserted into the router.

Just understand that if you type **interface e0** on a 2500, **interface fastethernet 0/0** on a 2600, or **interface serial 0/1/0** on a 2800, all you're doing is choosing an interface to configure, and basically, they're all configured the same way after that.

I'm going to continue with our router interface discussion in the next sections, and I'll include how to bring up the interface and set an IP address on a router interface.

Bringing Up an Interface

You can disable an interface with the interface command **shutdown** and enable it with the **no shutdown** command.

If an interface is shut down, it'll display administratively down when you use the **show interfaces** command (**sh int** for short):

```

Todd#sh int f0/1
FastEthernet0/1 is administratively down, line protocol is down
[output cut]

```

Another way to check an interface's status is via the **show running-config** command. All interfaces are shut down by default. You can bring up the interface with the **no shutdown** command (**no shut** for short):

```

Todd#config t
Todd(config)#int f0/1
Todd(config-if)#no shutdown
Todd(config-if)#
*Feb 28 22:45:08.455: %LINK-3-UPDOWN: Interface FastEthernet0/1,
    changed state to up
Todd(config-if)#do show int f0/1
FastEthernet0/1 is up, line protocol is up
[output cut]

```

Configuring an IP Address on an Interface

Even though you don't have to use IP on your routers, it's most often what people actually do use. To configure IP addresses on an interface, use the `ip address` command from interface configuration mode:

```
Todd(config)#int f0/1
Todd(config-if)#ip address 172.16.10.2 255.255.255.0
```

Don't forget to enable the interface with the `no shutdown` command. Remember to look at the command `show interface int` to see if the interface is administratively shut down or not. `show running-config` will also give you this information.



The `ip address address mask` command starts the IP processing on the interface.

If you want to add a second subnet address to an interface, you have to use the `secondary` parameter. If you type another IP address and press Enter, it will replace the existing IP address and mask. This is definitely a most excellent feature of the Cisco IOS.

So let's try it. To add a secondary IP address, just use the `secondary` parameter:

```
Todd(config-if)#ip address 172.16.20.2 255.255.255.0 ?
    secondary  Make this IP address a secondary address
    <cr>
Todd(config-if)#ip address 172.16.20.2 255.255.255.0 secondary
Todd(config-if)#^Z
Todd(config-if)#do sh run
Building configuration...
[output cut]
```

```
interface FastEthernet0/1
  ip address 172.16.20.2 255.255.255.0 secondary
  ip address 172.16.10.2 255.255.255.0
  duplex auto
  speed auto
!
```

I really wouldn't recommend having multiple IP addresses on an interface because it's ugly and inefficient, but I showed you anyway just in case you someday find yourself dealing with an MIS manager who's in love with really bad network design and makes you administer it! And who knows? Maybe someone will ask you about it someday and you'll get to seem really smart because you know this.

Configuring DHCP on your router

You can configure your router to become a Dynamic Host Configuration Protocol (DHCP) server, and I'll show you some examples at the end of this chapter and on a real network in Chapter 6 "IP Routing." Here are some basics of making your router a DHCP server.

- First, create a name for the pool. The name is completely irrelevant, but using a name allows you to create an unlimited amount of DHCP pools. A pool is defined as a subnet or group of IP addresses you want the DHCP server to hand out to DHCP clients.
- Second, use the **network** command to define the subnet of address the DHCP pool will hand out to the DHCP clients
- Third, set the optional options such as default gateway, DNS server, WINS server, domain name, and lease length. A DHCP lease specifies the amount of time a DHCP server grants a DHCP client permission to use a particular IP address.

Here is an example:

```
Todd#config t
Todd(config)#ip dhcp pool Finance
Todd(dhcp-config)#network 10.10.1.0 255.255.255.0
Todd(dhcp-config)#default-router 10.10.1.1
Todd(dhcp-config)#dns-server 10.30.1.1
Todd(dhcp-config)#netbios-name-server 10.30.1.1
Todd(dhcp-config)#lease 0 0 1
Todd(dhcp-config)#exit
Todd(config)#ip dhcp excluded-address 10.10.1.1 10.10.1.199
Todd(config)#
```

The above commands create a DHCP pool that will be used for DHCP clients connecting to a router interface with an IP address of 10.10.1.1 (the configured default gateway). The DNS and WINS server is 10.30.1.1 and the IP addresses that will be handed out to DHCP clients will be 10.10.1.200 through 10.10.1.254 because the excluded range is 10.10.1.1 through 10.10.1.199. The **lease** command is configured in days, hours, minutes. The **lease 0 0 1** command is a one-minute DHCP lease.

You need to verify and configure the router interface the DHCP clients are connected to with the correct IP address:

```
Todd(config)#interface f0/0
Todd(config-if)#ip address 10.10.1.1 255.255.255.0
Todd(config-if)#no shutdown
```

You can also configure a router interface to receive a DHCP address. You would do this, for example, if your router was connected to a DSL or cable connection to the

Internet. Here is an example on how you would configure a router interface to be a DHCP client:

```
Todd(config)#interface f0/0
Todd(config-if)#ip address dhcp
Todd(config-if)#no shutdown
```



Creating DHCP pools can be accomplished with the SDM as well.

Using the Pipe

No, not that pipe. I mean the output modifier. (Although with some of the router configurations I've seen in my career, sometimes I wonder!) This pipe (|) allows us to wade through all the configurations or other long outputs and get straight to our goods fast. Here's an example:

```
Todd#sh run | ?
  append    Append redirected output to URL (URLs supporting append operation
            only)
  begin      Begin with the line that matches
  exclude    Exclude lines that match
  include    Include lines that match
  redirect   Redirect output to URL
  section    Filter a section of output
  tee        Copy output to URL
```

```
Todd#sh run | begin interface
interface FastEthernet0/0
  description Sales VLAN
  ip address 10.10.10.1 255.255.255.248
  duplex auto
  speed auto
!
interface FastEthernet0/1
  ip address 172.16.20.2 255.255.255.0 secondary
  ip address 172.16.10.2 255.255.255.0
  duplex auto
  speed auto
!
```

```
interface Serial0/0/0
description Wan to SF circuit number 6fdda 12345678
no ip address
!
```

So basically, the pipe symbol (output modifier) is what you need to help you get where you want to go light years faster than mucking around in a router's entire configuration. I use it a lot when I am looking at a large routing table to find out whether a certain route is in the routing table. Here's an example:

```
Todd#sh ip route | include 192.168.3.32
R      192.168.3.32 [120/2] via 10.10.10.8, 00:00:25, FastEthernet0/0
Todd#
```

First, you need to know that this routing table had over 100 entries, so without my trusty pipe, I'd probably still be looking through that output! It's a powerfully efficient tool that saves you major time and effort by quickly finding a line in a configuration—or as the preceding example shows, a single route in a huge routing table.

Give yourself a little time to play around with the pipe command; get the hang of it, and you'll be seriously high on your newfound ability to quickly parse through router output.

Summarizing Some of Our Commands So Far.

Let's just use some simple terms and apply commands to them:

Enter global configuration mode: Router#**configure terminal**

Enter interface configuration mode : Router(config)#**interface s0/0**

Configure the interface IP address : Router(config-if)#**ip address 10.1.5.255 255.255.252.0**

Enable the interface: Router(config-if)#**no shutdown**

Label the interface: Router(config-if)#**description WAN to Miami**

Taking a look at the commands in order:

```
Router#config t
Router(config)#int s0/0
Router(config-if)#ip address 10.1.5.255 255.255.252.0
Router(config-if)#no shutdown
Router(config-if)#description WAN to Miami
```

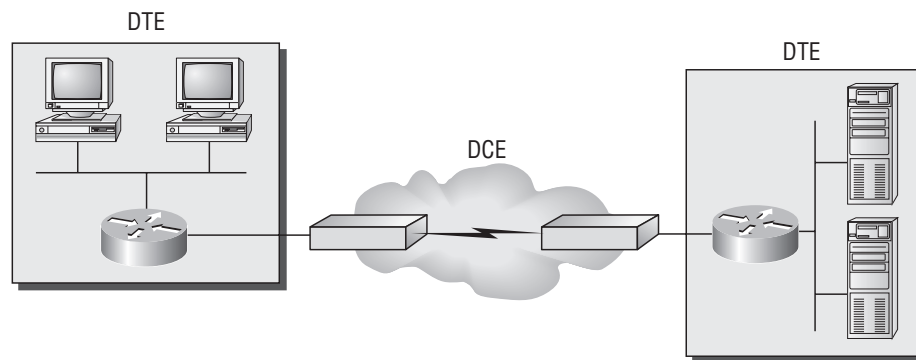
Serial Interface Commands

Wait! Before you just jump in and configure a serial interface, you need some key information—like knowing that the interface will usually be attached to a channel service unit/data service unit (CSU/DSU) type of device that provides clocking for the line to the router, as I've shown in Figure 4.5.

Here you can see that the serial interface is used to connect to a DCE network via a CSU/DSU that provides the clocking to the router interface. But if you have a back-to-back configuration (for example, one that's used in a lab environment like the one I've shown you in Figure 4.6), one end—the data communication equipment (DCE) end of the cable—must provide clocking!

By default, Cisco routers are all data terminal equipment (DTE) devices, which means that you must configure an interface to provide clocking if you need it to act like a DCE device. Again, you would not provide clocking on a production T1 connection, for example, because you would have a CSU/DSU connected to your serial interface, as Figure 4.5 shows.

FIGURE 4.5 A typical WAN connection

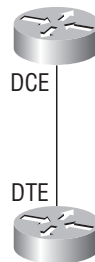


Clocking is typically provided by DCE network to routers.
In nonproduction environments, a DCE network is not always present.

FIGURE 4.6 Providing clocking on a nonproduction network

Set clock rate if needed.

```
Todd#config t
Todd(config)#interface serial 0
Todd(config-if)#clock rate 64000
```



DCE side determined by cable.
Add clocking to DCE side only.

show controllers will show the cable connection type.

You configure a DCE serial interface with the `clock rate` command:

```
Todd#config t
```

Enter configuration commands, one per line. End with CNTL/Z.

```
Todd(config)#int s0/0/0
```

```
Todd(config-if)#clock rate ?
```

Speed (bits per second)

```
1200  
2400  
4800  
9600  
14400  
19200  
28800  
32000  
38400  
48000  
56000  
57600  
64000  
72000  
115200  
125000  
128000  
148000  
192000  
250000  
256000  
384000  
500000  
512000  
768000  
800000  
1000000  
2000000  
4000000  
5300000  
8000000
```

<300-8000000> Choose clockrate from list above

```
Todd(config-if)#clock rate 1000000
```

The `clock rate` command is set in bits per second. Besides looking at the cable end to check for a label of DCE or DTE, you can see if a router's serial interface has a DCE cable connected with the `show controllers int` command:

```
Todd#sh controllers s0/0/0
Interface Serial0/0/0
Hardware is GT96K
DTE V.35idb at 0x4342FCB0, driver data structure at 0x434373D4
```

Here is an example of an output that shows a DCE connection:

```
Todd#sh controllers s0/2/0
Interface Serial0/2/0
Hardware is GT96K
DCE V.35, clock rate 1000000
```

The next command you need to get acquainted with is the `bandwidth` command. Every Cisco router ships with a default serial link bandwidth of T1 (1.544Mbps). But this has nothing to do with how data is transferred over a link. The bandwidth of a serial link is used by routing protocols such as EIGRP and OSPF to calculate the best cost (path) to a remote network. So if you're using RIP routing, the bandwidth setting of a serial link is irrelevant since RIP uses only hop count to determine that. If you're rereading this part thinking, "Huh—what? Routing protocols? Metrics?"—don't freak! I'm going over all that soon in Chapter 6.

Here's an example of using the `bandwidth` command:

```
Todd#config t
Todd(config)#int s0/0/0
Todd(config-if)#bandwidth ?
  <1-10000000> Bandwidth in kilobits
  inherit      Specify that bandwidth is inherited
  receive      Specify receive-side bandwidth
Todd(config-if)#bandwidth 1000
```

Did you notice that, unlike the `clock rate` command, the `bandwidth` command is configured in kilobits?



After going through all these configuration examples regarding the **clock rate** command, understand that the new ISR routers automatically detect DCE connections and set the clock rate to 2000000. However, you still need to understand the **clock rate** command, even though the new routers set it for you automatically!

Viewing, Saving, and Erasing Configurations

If you run through setup mode, you'll be asked if you want to use the configuration you just created. If you say yes, then it will copy the configuration running in DRAM (known as the running-config) into NVRAM and name the file startup-config. Hopefully, you always will use the CLI or SDM and not setup mode.

You can manually save the file from DRAM to NVRAM by using the `copy running-config startup-config` command (you can use the shortcut `copy run start`):

```
Todd#copy running-config startup-config
Destination filename [startup-config]? [press enter]
Building configuration...
[OK]
Todd#
Building configuration...
```

When you see a question with an answer in [], it means that if you just press Enter, you're choosing the default answer.

Also, when the command asks for the destination filename, the default answer is startup-config. The reason it asks is because you can copy the configuration pretty much anywhere you want. Take a look:

```
Todd#copy running-config ?
archive:      Copy to archive: file system
flash:       Copy to flash: file system
ftp:         Copy to ftp: file system
http:        Copy to http: file system
https:       Copy to https: file system
ips-sdf      Update (merge with) IPS signature configuration
null:       Copy to null: file system
nvram:       Copy to nvram: file system
rcp:        Copy to rcp: file system
running-config Update (merge with) current system configuration
scp:        Copy to scp: file system
startup-config Copy to startup configuration
syslog:     Copy to syslog: file system
system:     Copy to system: file system
tftp:       Copy to tftp: file system
xmodem:     Copy to xmodem: file system
ymodem:     Copy to ymodem: file system
```

We'll take a closer look at how and where to copy files in Chapter 5.

You can view the files by typing **show running-config** or **show startup-config** from privileged mode. The **sh run** command, which is a shortcut for **show running-config**, tells us that we are viewing the current configuration:

```
Todd#show running-config
Building configuration...

Current configuration : 3343 bytes
!
version 12.4
[output cut]
```

The **sh start** command—one of the shortcuts for the **show startup-config** command—shows us the configuration that will be used the next time the router is reloaded. It also tells us how much NVRAM is being used to store the startup-config file. Here's an example:

```
Todd#show startup-config
Using 1978 out of 245752 bytes
!
version 12.4
[output cut]
```

Deleting the Configuration and Reloading the Router

You can delete the startup-config file by using the **erase startup-config** command:

```
Todd#erase startup-config
Erasing the nvram filesystem will remove all configuration files!
Continue? [confirm][enter]
[OK]
Erase of nvram: complete
Todd#
*Feb 28 23:51:21.179: %SYS-7-NV_BLOCK_INIT: Initialized the geometry of nvram
Todd#sh startup-config
startup-config is not present
Todd#reload
Proceed with reload? [confirm]System configuration has been modified.
Save? [yes/no]: n
```

If you reload or power down and up the router after using the `erase startup-config` command, you'll be offered setup mode because there's no configuration saved in NVRAM. You can press Ctrl+C to exit setup mode at any time (the `reload` command can only be used from privileged mode).

At this point, you shouldn't use setup mode to configure your router. So just say no to setup mode, because it's there to help people who don't know how to use the Cash Line Interface (CLI), and this no longer applies to you. Be strong—you can do it!

Verifying Your Configuration

Obviously, `show running-config` would be the best way to verify your configuration and `show startup-config` would be the best way to verify the configuration that'll be used the next time the router is reloaded—right?

Well, once you take a look at the running-config, if all appears well, you can verify your configuration with utilities such as Ping and Telnet. Ping is Packet Internet Groper, a program that uses ICMP echo requests and replies. (ICMP is discussed in Chapter 2, "Internet Protocols.") Ping sends a packet to a remote host, and if that host responds, you know that it's alive. But you don't know if it's alive and also *well*—just because you can ping a Microsoft server does not mean you can log in! Even so, Ping is an awesome starting point for troubleshooting an internetwork.

Did you know that you can ping with different protocols? You can, and you can test this by typing **ping ?** at either the router user-mode or privileged-mode prompt:

```
Router#ping ?
WORD      Ping destination address or hostname
appletalk Appletalk echo
clns      CLNS echo
decnet    DECnet echo
ip        IP echo
ipv6      IPv6 echo
ipx       Novell/IPX echo
srb       srb echo
tag       Tag encapsulated IP echo
<cr>
```

If you want to find a neighbor's Network layer address, either you need to go to the router or switch itself or you can type **show cdp entry * protocol** to get the Network layer addresses you need for ping.



Cisco Discovery Protocol (CDP) is covered in Chapter 5.

Traceroute uses ICMP with IP time to live (TTL) time-outs to track the path a packet takes through an internetwork, in contrast to Ping, which just finds the host and responds. And Traceroute can also be used with multiple protocols.

Router#**traceroute ?**

```
WORD          Trace route to destination address or hostname
appletalk     AppleTalk Trace
clns          ISO CLNS Trace
ip            IP Trace
ipv6          IPv6 Trace
ipx           IPX Trace
<cr>
```

Telnet, FTP, and HTTP are really the best tools because they use IP at the Network layer and TCP at the Transport layer to create a session with a remote host. If you can telnet, ftp, or http into a device, your IP connectivity just has to be good.

Router#**telnet ?**

```
WORD IP address or hostname of a remote system
<cr>
```

From the router prompt, you just type a hostname or IP address and it will assume you want to telnet—you don't need to type the actual command, **telnet**.

In the following sections, I am going to show you how to verify the interface statistics.

Verifying with the *show interface* Command

Another way to verify your configuration is by typing **show interface** commands, the first of which is **show interface ?**. That will reveal all the available interfaces to configure.



The **show interfaces** command displays the configurable parameters and statistics of all interfaces on a router.

This command is very useful for verifying and troubleshooting router and network issues. The following output is from my freshly erased and rebooted 2811 router:

Router#**sh int ?**

```
Async          Async interface
BVI            Bridge-Group Virtual Interface
CDMA-Ix        CDMA Ix interface
CTunnel        CTunnel interface
Dialer         Dialer interface
FastEthernet   FastEthernet IEEE 802.3
Loopback       Loopback interface
MFR            Multilink Frame Relay bundle interface
```

Multilink	Multilink-group interface
Null	Null interface
Port-channel	Ethernet Channel of interfaces
Serial	Serial
Tunnel	Tunnel interface
Vif	PGM Multicast Host interface
Virtual-PPP	Virtual PPP interface
Virtual-Template	Virtual Template interface
Virtual-TokenRing	Virtual TokenRing
accounting	Show interface accounting
counters	Show interface counters
crb	Show interface routing/bridging info
dampening	Show interface dampening info
description	Show interface description
etherchannel	Show interface etherchannel information
irb	Show interface routing/bridging info
mac-accounting	Show interface MAC accounting info
mpls-exp	Show interface MPLS experimental accounting info
precedence	Show interface precedence accounting info
pruning	Show interface trunk VTP pruning information
rate-limit	Show interface rate-limit info
stats	Show interface packets & octets, in & out, by switching path
status	Show interface line status
summary	Show interface summary
switching	Show interface switching
switchport	Show interface switchport information
trunk	Show interface trunk information
	Output modifiers
<cr>	

The only “real” physical interfaces are FastEthernet, serial, and async; the rest are all logical interfaces or commands to verify with.

The next command is `show interface fastethernet 0/0`. It reveals to us the hardware address, logical address, and encapsulation method as well as statistics on collisions, as seen here:

Router#sh int f0/0

FastEthernet0/0 is up, line protocol is up

Hardware is MV96340 Ethernet, address is 001a.2f55.c9e8 (bia 001a.2f55.c9e8)

Internet address is 192.168.1.33/27

MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,

reliability 255/255, txload 1/255, rxload 1/255

```

Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
Auto-duplex, Auto Speed, 100BaseTX/FX
ARP type: ARPA, ARP Timeout 04:00:00
Last input never, output 00:02:07, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
Queueing strategy: fifo
Output queue: 0/40 (size/max)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
  0 packets input, 0 bytes
    Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
  0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
  0 watchdog
  0 input packets with dribble condition detected
16 packets output, 960 bytes, 0 underruns
  0 output errors, 0 collisions, 0 interface resets
  0 babbles, 0 late collision, 0 deferred
  0 lost carrier, 0 no carrier
  0 output buffer failures, 0 output buffers swapped out
Router#

```

As you probably guessed, we're going to discuss the important statistics from this output, but first, for fun (this is all fun, right?), I've got to ask you, What subnet is the FastEthernet 0/0 a member of and what's the broadcast address and valid host range?

And, my friend, you really have to be able to nail these things NASCAR fast! Just in case you didn't, the address is 192.168.1.33/27. And I've gotta be honest—if you don't know what a /27 is at this point, you'll need a miracle to pass the exam. (A /27 is 255.255.255.224.) The fourth octet is a block size of 32. The subnets are 0, 32, 64, ...; the FastEthernet interface is in the 32 subnet; the broadcast address is 63; and the valid hosts are 33–62.



If you struggled with any of this, please save yourself from certain doom and get yourself back into Chapter 3 now! Read and reread it until you've got it dialed in!

The preceding interface is working and looks to be in good shape. The `show interfaces` command will show you if you are receiving errors on the interface, and it will show you the maximum transmission units (MTUs), bandwidth (BW), reliability (255/255 means perfect!), and load (1/255 means no load).

Continuing to use the previous output, what is the bandwidth of the interface? Well, other than the easy giveaway of the interface being called a FastEthernet interface, we can see that

the bandwidth is 100000Kbit, which is 100,000,000 (Kbit means to add three zeros), which is 100Mbps per second, or FastEthernet. Gigabit would be 1000000Kbits per second.

The most important statistic of the `show interface` command is the output of the line and data-link protocol status. If the output reveals that FastEthernet 0/0 is up and the line protocol is up, then the interface is up and running:

```
Router#sh int fa0/0
FastEthernet0/0 is up, line protocol is up
```

The first parameter refers to the Physical layer, and it's up when it receives carrier detect. The second parameter refers to the Data Link layer, and it looks for keepalives from the connecting end. (Keepalives are used between devices to make sure connectivity has not dropped.)

Here's an example of where the problem usually is found—on serial interfaces:

```
Router#sh int s0/0/0
Serial0/0 is up, line protocol is down
```

If you see that the line is up but the protocol is down, as shown above, you're experiencing a clocking (keepalive) or framing problem—possibly an encapsulation mismatch. Check the keepalives on both ends to make sure that they match; that the clock rate is set, if needed, and that the encapsulation type is the same on both ends. The output above would be considered a Data Link layer problem.

If you discover that both the line interface and the protocol are down, it's a cable or interface problem. The following output would be considered a Physical layer problem:

```
Router#sh int s0/0/0
Serial0/0 is down, line protocol is down
```

If one end is administratively shut down (as shown next), the remote end would present as down and down:

```
Router#sh int s0/0/0
Serial0/0 is administratively down, line protocol is down
```

To enable the interface, use the command `no shutdown` from interface configuration mode.

The next `show interface serial 0/0/0` command demonstrates the serial line and the maximum transmission unit (MTU)—1,500 bytes by default. It also shows the default bandwidth (BW) on all Cisco serial links: 1.544Kbps. This is used to determine the bandwidth of the line for routing protocols such as EIGRP and OSPF. Another important configuration to notice is the keepalive, which is 10 seconds by default. Each router sends a keepalive message to its neighbor every 10 seconds, and if both routers aren't configured for the same keepalive time, it won't work.

```
Router#sh int s0/0/0
Serial0/0 is up, line protocol is up
Hardware is HD64570
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
```

```

Encapsulation HDLC, loopback not set, keepalive set
(10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Queueing strategy: fifo
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
  0 packets input, 0 bytes, 0 no buffer
    Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
  0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored,
  0 abort
  0 packets output, 0 bytes, 0 underruns
  0 output errors, 0 collisions, 16 interface resets
  0 output buffer failures, 0 output buffers swapped out
  0 carrier transitions
  DCD=down DSR=down DTR=down RTS=down CTS=down

```

You can clear the counters on the interface by typing the command **clear counters**:

Router#**clear counters ?**

Async	Async interface
BVI	Bridge-Group Virtual Interface
CTunnel	CTunnel interface
Dialer	Dialer interface
FastEthernet	FastEthernet IEEE 802.3
Group-Async	Async Group interface
Line	Terminal line
Loopback	Loopback interface
MFR	Multilink Frame Relay bundle interface
Multilink	Multilink-group interface
Null	Null interface
Serial	Serial
Tunnel	Tunnel interface
Vif	PGM Multicast Host interface
Virtual-Template	Virtual Template interface
Virtual-TokenRing	Virtual TokenRing

<cr>

Router#**clear counters s0/0/0**

Clear "show interface" counters on this interface
 [confirm][enter]

```
Router#
00:17:35: %CLEAR-5-COUNTERS: Clear counter on interface
Serial0/0/0 by console
Router#
```

Verifying with the *show ip interface* Command

The `show ip interface` command will provide you with information regarding the layer 3 configurations of a router's interfaces:

```
Router#sh ip interface
FastEthernet0/0 is up, line protocol is up
Internet address is 1.1.1.1/24
Broadcast address is 255.255.255.255
Address determined by setup command
MTU is 1500 bytes
Helper address is not set
Directed broadcast forwarding is disabled
Outgoing access list is not set
Inbound access list is not set
Proxy ARP is enabled
Security level is default
Split horizon is enabled
[output cut]
```

The status of the interface, the IP address and mask, information on whether an access list is set on the interface, and basic IP information are included in this output.

Using the *show ip interface brief* Command

The `show ip interface brief` command is probably one of the most helpful commands that you can ever use on a Cisco router. This command provides a quick overview of the router's interfaces, including the logical address and status:

```
Router#sh ip int brief
```

Interface	IP-Address	OK?	Method	Status	Protocol
FastEthernet0/0	unassigned	YES	unset	up	up
FastEthernet0/1	unassigned	YES	unset	up	up
Serial0/0/0	unassigned	YES	unset	up	down
Serial0/0/1	unassigned	YES	unset	administratively down	down
Serial0/1/0	unassigned	YES	unset	administratively down	down
Serial0/2/0	unassigned	YES	unset	administratively down	down

Remember, administratively down means that you need to type **no shutdown** under the interface. Notice that Serial0/0/0 is up/down, which means that the physical layer is good and carrier detect is sensed but no keepalives are being received from the remote end. In a nonproduction network, like the one I am working with, the clock rate isn't set.

Verifying with the *show protocols* Command

The `show protocols` command is a really helpful command you'd use in order to quickly see the status of layers 1 and 2 of each interface as well as the IP addresses used.

Here's a look at one of my production routers:

```
Router#sh protocols
```

```
Global values:
```

```
Internet Protocol routing is enabled
Ethernet0/0 is administratively down, line protocol is down
Serial0/0 is up, line protocol is up
Internet address is 100.30.31.5/24
Serial0/1 is administratively down, line protocol is down
Serial0/2 is up, line protocol is up
Internet address is 100.50.31.2/24
Loopback0 is up, line protocol is up
Internet address is 100.20.31.1/24
```

Using the *show controllers* Command

The `show controllers` command displays information about the physical interface itself. It'll also give you the type of serial cable plugged into a serial port. Usually, this will only be a DTE cable that plugs into a type of data service unit (DSU).

```
Router#sh controllers serial 0/0
```

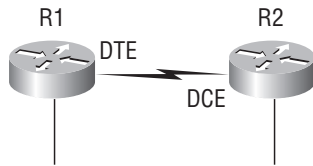
```
HD unit 0, idb = 0x1229E4, driver structure at 0x127E70
buffer size 1524 HD unit 0, V.35 DTE cable
cpb = 0xE2, eda = 0x4140, cda = 0x4000
```

```
Router#sh controllers serial 0/1
```

```
HD unit 1, idb = 0x12C174, driver structure at 0x131600
buffer size 1524 HD unit 1, V.35 DCE cable
cpb = 0xE3, eda = 0x2940, cda = 0x2800
```

Notice that serial 0/0 has a DTE cable, whereas the serial 0/1 connection has a DCE cable. Serial 0/1 would have to provide clocking with the `clock rate` command. Serial 0/0 would get its clocking from the DSU.

Let's look at this command again. In Figure 4.7, see the DTE/DCE cable between the two routers? Know that you will not see this in production networks!

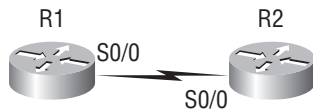
FIGURE 4.7 The show controllers command

Router R1 has a DTE connection—the default for all Cisco routers. Routers R1 and R2 can't communicate. Check out the output of the `show controllers s0/0` command here:

```
R1#sh controllers serial 0/0
HD unit 0, idb = 0x1229E4, driver structure at 0x127E70
buffer size 1524 HD unit 0, V.35 DCE cable
cpb = 0xE2, eda = 0x4140, cda = 0x4000
```

The `show controllers s0/0` command shows that the interface is a V.35 DCE cable. This means that R1 needs to provide clocking of the line to router R2. Basically, the interface has the wrong label on the cable on the R1 router's serial interface. But if you add clocking on the R1 router's serial interface, the network should come right up.

Let's check out another issue, shown in Figure 4.8, that you can solve by using the `show controllers` command. Again, routers R1 and R2 can't communicate.

FIGURE 4.8 The show controllers command used with the show ip interface command

Here's the output of R1's `show controllers s0/0` command and `show ip interface s0/0`:

```
R1#sh controllers s0/0
HD unit 0, idb = 0x1229E4, driver structure at 0x127E70
buffer size 1524 HD unit 0,
DTE V.35 clocks stopped
cpb = 0xE2, eda = 0x4140, cda = 0x4000
```

```
R1#sh ip interface s0/0
Serial0/0 is up, line protocol is down
Internet address is 192.168.10.2/24
Broadcast address is 255.255.255.255
```

If you use the `show controllers` command and the `show ip interface` command, you'll see that router R1 isn't receiving clocking of the line. This network is a nonproduction network,

so no CSU/DSU is connected to provide clocking of the line. This means the DCE end of the cable will be providing the clock rate—in this case, the R2 router. The `show ip interface` indicates that the interface is up but the protocol is down, which means that no keepalives are being received from the far end. In this example, the likely culprit is the result of bad cable, or no clocking.

Cisco's Security Device Manager (SDM)

So finally, here it is...the part about Cisco's SDM. This beauty is used to help you configure a router from an HTTP or HTTPS interface. Actually, to be real, Cisco did produce something like this in the past, but honestly, it just didn't work all that well. This one does—really! And by the way, SDM is available on Cisco router models from Cisco 830 Series to 7301. Plus, it's preinstalled on all new 850, 870, 1800, 2800, and 3800 series routers. It's all good!

But there is a catch. Although the SDM is a really great tool, it's best if you pull it out just for advanced configurations—not the super-small, simple configurations like those we used in this chapter. Let me explain. Let's say you wanted to set up an advanced access list, VPN with IPsec, and intrusion protection on your router. If so, then SDM is your baby. You don't even have to know what IPsec means to configure it and make it work.

But again, there's both good news and bad news with this. Of course, we can now handle advanced configurations much more easily, but at the same time, it's not so good because, well, literally anyone can do the same thing!

What I'm going to do is show you how to log in using SDM; set your hostname, banner, and enable secret password; and assign a DHCP pool on a router and an IP address to an interface. And if it all goes well, after this section you'll find that SDM will turn out to be much more of a good thing because it will prove to be so much more helpful to you when you're reading the more advanced chapters on the IOS, NAT, wireless technologies, and security in both this book and the CCNA study guide than the way it used to be when we were all mired down in basic IOS router configuration. I'll use it a little in every chapter, both to show you the ease you'll gain and to explain its complexity. But I promise to truly expose the limitations of SDM as well.

Honestly, I could write a whole book about SDM, but I don't need to because Cisco did it already. To find more details about SDM, go to www.cisco.com/go/sdm. Plus, a new router will typically come with a CD that walks you through physically connecting your router and configuring it step-by-step (overkill in my opinion); you truly don't need this CD to connect to your router and use SDM.

All you need is a supported ISR router (1800/2800, etc.) and you can download the latest version of SDM complete with instructions for installing it on your computer and your router from the following location: www.cisco.com/cgi-bin/tablebuild.pl/sdm.



To download this software, you need a Cisco Connection Online (CCO) login. However, this is free and only takes a minute or less to set up. Go to www.cisco.com and click on Register in the top-right corner. Fill in the short form, add the username and password you want to use, and you can now download the SDM and the SDM demo.

From this site, you can not only install the SDM on your computer to help the SDM pages load faster when connecting to your router, you can also enable the use of the Cisco SDM demo itself.



I highly recommend downloading and running through the SDM demo if you don't have an ISR router to play with. First, install SDM on your computer, and then download the demo at www.cisco.com/cgi-bin/tablebuild.pl/sdm-tool-demo. Read the install instructions, or run through Hands-on Lab 4.6 at the end of this chapter to get you set up and going strong with SDM.

You need to know this though—to set up your host to log in using the SDM, you have to make sure your router is configured first. In the last CLI section of this chapter, I had deleted my configurations and reloaded the router, so I had to start from scratch. But doing that really isn't all that hard. You can just choose a LAN interface of the router and then connect a host directly to the router using a crossover cable—not so bad!

```
Would you like to enter the initial configuration dialog? [yes/no]: n
```

```
Press RETURN to get started!
```

```
Router>en
```

```
Router#config t
```

```
Enter configuration commands, one per line. End with CNTL/Z.
```

```
Router(config)#int f0/0
```

```
Router(config-if)#ip address 1.1.1.1 255.255.255.0
```

```
Router(config-if)#no shut
```

```
Router(config-if)#do ping 1.1.1.2
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 1.1.1.2, timeout is 2 seconds:
```

```
!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

So what's this all about? In the preceding configuration, I configured the FastEthernet interface with an IP address and enabled it with the `no shutdown` command. I then tested my connection by pinging my directly connected host from the router prompt. (This would be a minimum configuration and will allow you to connect via SDM.) From here, you just open a browser, with pop-ups enabled, type **http://1.1.1.1**, and follow the easy prompts once connected.

It's different if you want to set the router to use HTTPS, which allows you access into privileged mode upon connection (meaning we're setting the router back to the original default configuration). And you need to add a few more commands.

First, enable the HTTP/HTTPS server (your router won't support HTTPS if it doesn't have the advanced services IOS):

```
Router(config)#ip http server
```

```
Router(config)#ip http secure-server
```

```
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
```

```
Router(config)#ip http authentication local
```

Second, create a user account using privilege level 15 (the highest level):

```
Router(config)#username cisco privilege ?
```

```
<0-15> User privilege level
```

```
Router(config)#username cisco privilege 15 password ?
```

```
0 Specifies an UNENCRYPTED password will follow
```

```
7 Specifies a HIDDEN password will follow
```

```
LINE The UNENCRYPTED (cleartext) user password
```

```
Router(config)#username cisco privilege 15 password 0 cisco
```

Last, configure the console, Telnet and SSH (if you want to use both Telnet and SSH) to provide local login authentication at privilege level access:

```
Router(config)#line console 0
```

```
Router(config-line)#login local
```

```
Router(config-line)#exit
```

```
Router(config)#line vty 0 ?
```

```
<1-1180> Last Line number
```

```
<cr>
```

```
Router(config)#line vty 0 1180
```

```
Router(config-line)#privilege level 15
```

```
Router(config-line)#login local
```

```
Router(config-line)#transport input telnet ssh
```

```
Router(config-line)#^Z
```

Voilà! Now let's connect to my 2811 using HTTPS!

Okay—as soon as I connected via `https://1.1.1.1`, I received a security alert message.



I was then prompted to log in with the username/password I created. SDM started loading and told me to hold on, which means it needs a bit of time to load another window. *Do not* close this window.



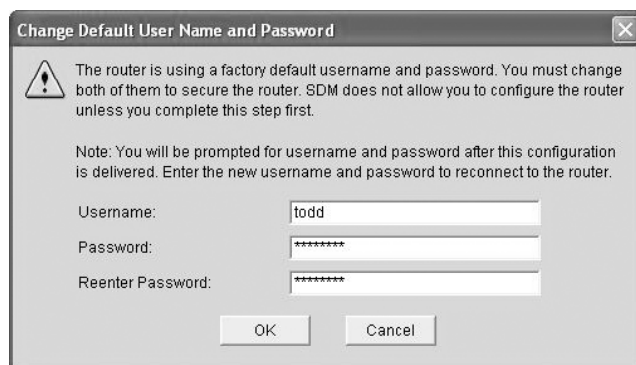
The certificate that I created with the `ip http-secure-server` command was loaded onto the router. I chose to click on Always Trust Content from this Publisher and then clicked Yes.



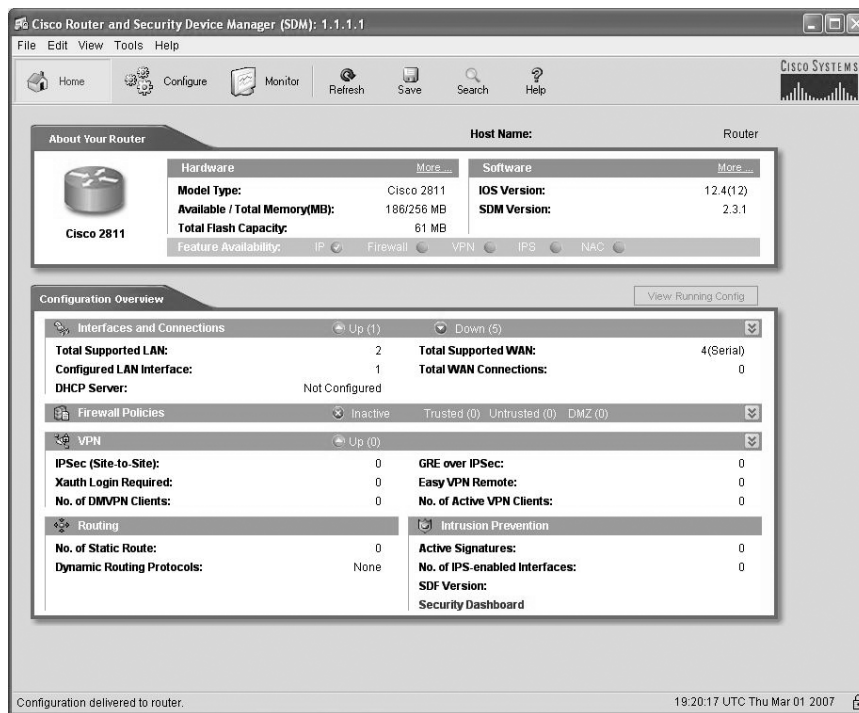
Of course, the certificate would not match any site name, so I then had to verify that I wanted to run it.



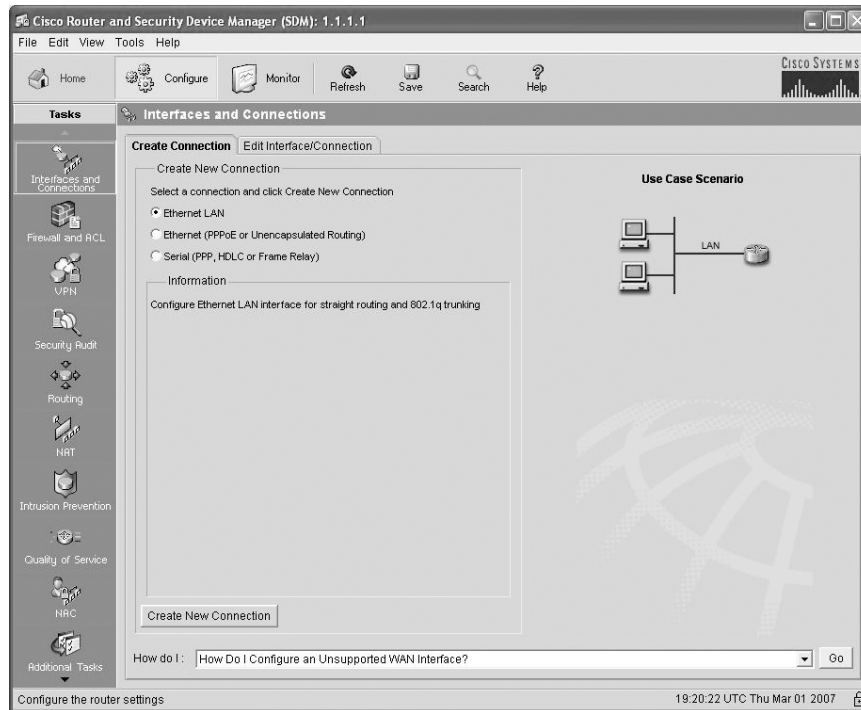
I then had to log in again and wait for SDM to load, during which time the router had me change my default username and password.



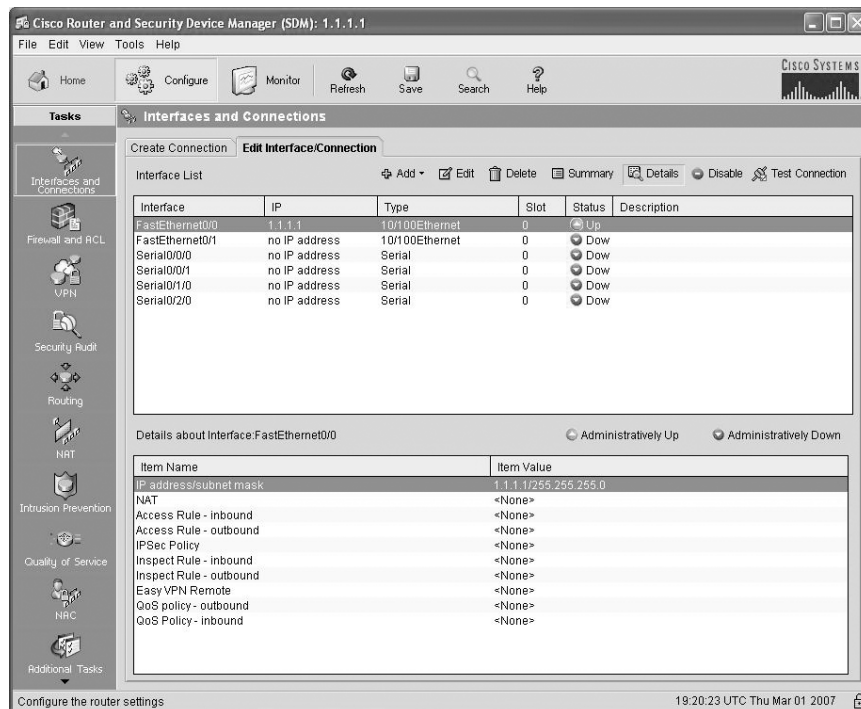
Finally—yes! I am connected to SDM!



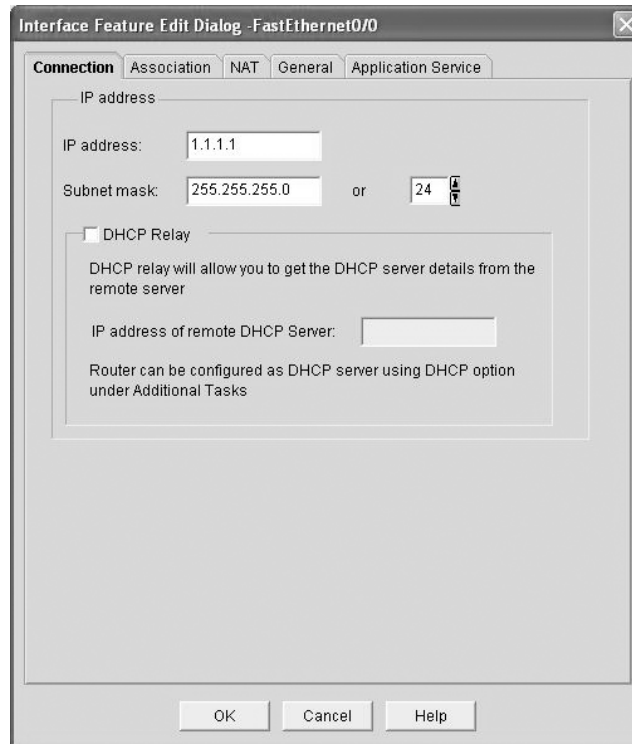
Clicking the Configure button at the top of the page, I chose to go step-by-step through interface configuration by first choosing the type of interface I wanted to configure and then clicking the Create New Connection button at the bottom of the page. This opens the LAN or WAN wizard, depending on which interface you choose to configure. (We'll configure our router interfaces with these wizards in Chapter 6.)



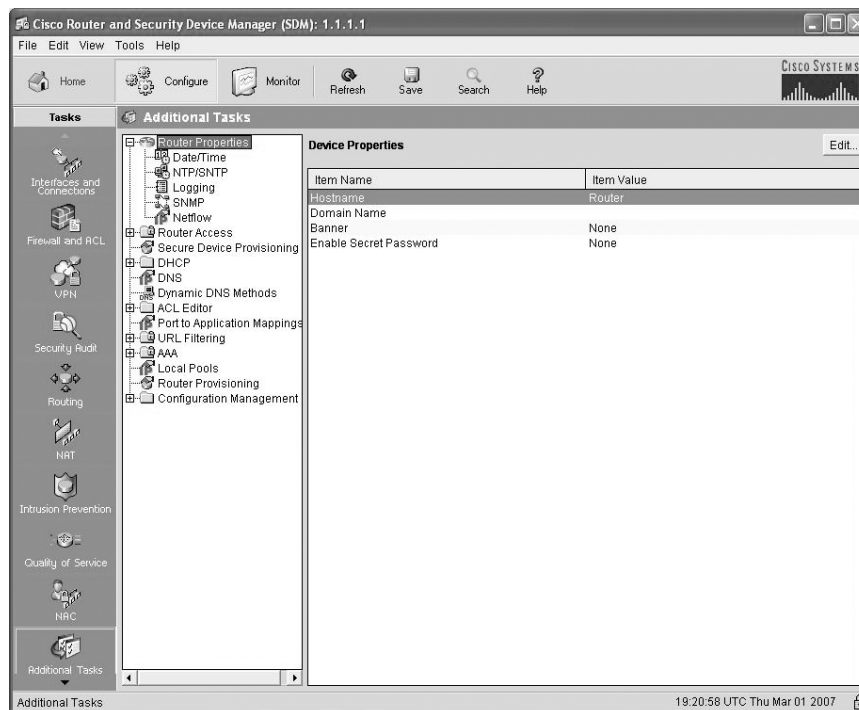
By clicking the Edit Interface/Connection tab, you get to see your interface status.



That's not all—just double-click on an interface to edit it. (You can only do this after you've gone through the LAN or WAN wizard and configured the interface.)



Look down at the bottom-left portion of the wizard page and click the Additional Tasks button. From there, just click the Router Properties icon.



Here, you can set the hostname, MOTD banner, and enable secret password. Last, I clicked on the DHCP folder, then the DHCP pool icon. I then clicked Add and created a DHCP pool on my router.

Now, let's take a look at the configuration on the router:

```
Todd#sh run
Building configuration...
[output cut]
hostname Todd
!
ip domain name lammle.com
[output cut]
ip dhcp excluded-address 172.16.10.1
ip dhcp excluded-address 172.16.10.11 172.16.10.254
!
ip dhcp pool Todd's_LAN
    import all
    network 172.16.10.0 255.255.255.0
!
crypto pki trustpoint TP-self-signed-2645776477
    enrollment selfsigned
    subject-name cn=IOS-Self-Signed-Certificate-2645776477
    revocation-check none
    rsakeypair TP-self-signed-2645776477
!
```

```

crypto pki certificate chain TP-self-signed-2645776477
certificate self-signed 01
  3082023E 308201A7 A0030201 02020101 300D0609 2A864886 F70D0101
  04050030 31312F30 2D060355 04031326 494F532D 53656C66 2D536967
  6E65642D 43657274 69666963 6174652D 32363435 37373634 3737301E
  170D3037 30333031 3139313 33335A17 0D323030 31303130 30303030
  305A3031 312F302D 06035504 03132649 4F532D53 656C662D 5369676E
  65642D43 65727469 66696361 74652D32 36343537 37363437 3730819F
  300D0609 2A864886 F70D0101 01050003 818D0030 81890281 8100BB24
  [output cut]
quit
username todd privilege 15 secret 5 $1$nvgs$QRNCWKJ7rfmtNNkD2xvGq/
[output cut]
!
line con 0
  login local
line aux 0
line vty 0 4
  privilege level 15
  login local
  transport input telnet ssh
line vty 5 1180
  privilege level 15
  login local
  transport input telnet ssh
!

```

So what we can see here is that the router created the hostname, DHCP pool, and certificate. We went through a lot of setup because we were using HTTPS—it's so much easier and unencumbered with less setup to just use HTTP. But remember, what we used was an ISR default configuration to connect using SDM. We'll continue to use SDM throughout this book, but I seriously encourage you to get SDM for yourself and start getting familiar with it!

Summary

This was a fun chapter! I showed you a lot about the Cisco IOS and I really hope you gained a lot of insight into the Cisco router world. This chapter started off by explaining the Cisco Internet-work Operating System (IOS) and how you can use the IOS to run and configure Cisco routers. You learned how to bring a router up and what setup mode does. Oh, and by the way, since you can now basically configure Cisco routers, you should never use setup mode, right?

After I discussed how to connect to a router with a console and LAN connection, I covered the Cisco help features and how to use the CLI to find commands and command parameters. In addition, I discussed some basic **show** commands to help you verify your configurations.

Administrative functions on a router help you administer your network and verify that you are configuring the correct device. Setting router passwords is one of the most important configurations you can perform on your routers. I showed you the five passwords to set. In addition, I used the hostname, interface description, and banners to help you administer your router.

Last, I showed you how to configure the router so you can connect using the Secure Device Manager and configure your router. Again, it's a whole lot easier to use the CLI to configure basic router functions, but I'll show you how SDM can really help with the more advanced configurations coming up soon.

Well, that concludes your introduction to the Cisco IOS. And, as usual, it's super-important for you to have the basics that we went over in this chapter before you move on to the following chapters.

Exam Essentials

Understand what happens (and the sequence in which it happens) when you power on a router.

When you first bring up a Cisco router, it will run a power-on self-test (POST), and if that passes, it will look for and load the Cisco IOS from flash memory, if a file is present. The IOS then proceeds to load and looks for a valid configuration in NVRAM called the startup-config. If no file is present in NVRAM, the router will go into setup mode.

Remember what setup mode provides. Setup mode is automatically started if a router boots and no startup-config is in NVRAM. You can also bring up setup mode by typing **setup** from privileged mode. Setup provides a minimum amount of configuration in an easy format for someone who does not understand how to configure a Cisco router from the command line.

Understand the difference between user mode and privileged mode. User mode provides a command-line interface with very few available commands by default. User mode does not allow the configuration to be viewed or changed. Privileged mode allows a user to both view and change the configuration of a router. You can enter privileged mode by typing the command **enable** and entering the enable password or enable secret password, if set.

Remember what the command **show version provides.** The **show version** command will provide basic configuration for the system hardware as well as the software version, the names and sources of configuration files, the **config-register** setting, and the boot images.

Remember how to set the hostname of a router. The command sequence to set the hostname of a router is as follows:

```
enable
config t
hostname Todd
```

Remember the difference between the enable password and enable secret password. Both of these passwords are used to gain access into privileged mode. However, the enable secret password is newer and is always encrypted by default. Also, if you set the enable password and then set the enable secret, only the enable secret will be used.

Remember how to set the enable secret on a router. To set the enable secret, you use the command `enable secret`. Do not use `enable secret password password` or you will set your password to *password password*. Here is an example:

```
enable
config t
enable secret todd
```

Remember how to set the console password on a router. To set the console password, use the following sequence:

```
enable
config t
line console 0
login
password todd
```

Remember how to set the Telnet password on a router. To set the Telnet password, the sequence is as follows:

```
enable
config t
line vty 0 4
password todd
login
```

Remember how to set up SSH on a router. To set a router to use SSH instead of, or with, Telnet, use the following steps:

1. Set your hostname:
Router(config)#hostname Todd
2. Set the domain name (both the hostname and domain name are required for the encryption keys to be generated):
Todd(config)#ip domain-name Lammle.com
3. Generate the encryption keys for securing the session:
Todd(config)#crypto key generate rsa general-keys modulus ?
4. <360-2048> size of the key modulus [360-2048]
Todd(config)#crypto key generate rsa general-keys modulus 1024

5. The name for the keys will be: Todd.Lammle.com
 % The key modulus size is 1024 bits
6. % Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
 *June 24 19:25:30.035: %SSH-5-ENABLED: SSH 1.99 has been enabled
7. Connect to the VTY lines of the router:
 Todd(config)#**line vty 0 1180**
8. Last, configure SSH as the access protocol:
 Todd(config-line)#**transport input ssh**

Understand how to troubleshoot a serial link problem. If you type **show interface serial 0** and see down, line protocol is down, this will be considered a Physical layer problem. If you see it as up, line protocol is down, then you have a Data Link layer problem.

Understand how to verify your router with the show interfaces command. If you type **show interfaces**, you can view the statistics for the interfaces on the router, verify whether the interfaces are shut down, and see the IP address of each interface.

Written Lab 4

Write out the command or commands for the following questions:

1. What command is used to set a serial interface to provide clocking to another router at 64Kb?
2. If you telnet into a router and get the response **connection refused**, **password not set**, what would you do on the destination router to stop receiving this message and not be prompted for a password?
3. If you type **show inter et 0** and notice the port is administratively down, what would you do?
4. If you wanted to delete the configuration stored in NVRAM, what would you type?
5. If you wanted to set a user-mode password for the console port, what would you type?
6. If you wanted to set the enable secret password to *cisco*, what would you type?
7. If you wanted to see if a serial interface needed to provide clocking, what command would you use?
8. What command would you use to see the terminal history size?
9. You want to reinitialize the router and totally replace the running-config with the current startup-config. What command will you use?
10. How would you set the name of a router to *Chicago*?

(The answers to Written Lab 4 can be found following the answers to the review questions for this chapter.)

Hands-on Labs

In this section, you will perform commands on a Cisco router that will help you understand what you learned in this chapter.

You'll need at least one Cisco router—two would be better, three would be outstanding. The hands-on labs in this section are included for use with real Cisco routers. If you are using software from RouterSim.com or Sybex, please use the hands-on labs found in those programs. It doesn't matter what series type router you use with these labs (i.e., 2500, 2600, 800, 1800, or 2800).

This chapter includes the following six labs:

Lab 4.1: Logging into a Router

Lab 4.2: Using the Help and Editing Features

Lab 4.3: Saving a Router Configuration

Lab 4.4: Setting Your Passwords

Lab 4.5: Setting the Hostname, Descriptions, IP Address, and Clock Rate

Lab 4.6: Installing SDM on Your Computer

Hands-on Lab 4.1: Logging into a Router

1. Press Enter to connect to your router. This will put you into user mode.
2. At the **Router>** prompt, type a question mark (?).
3. Notice the **-more-** at the bottom of the screen.
4. Press the Enter key to view the commands line by line. Press the spacebar to view the commands a full screen at a time. You can type **q** at any time to quit.
5. Type **enable** or **en** and press Enter. This will put you into privileged mode where you can change and view the router configuration.
6. At the **Router#** prompt, type a question mark (?). Notice how many options are available to you in privileged mode.
7. Type **q** to quit.
8. Type **config** and press Enter.
9. Press Enter to configure your router using your terminal.
10. At the **Router(config)#** prompt, type a question mark (?), then **q** to quit, or hit the spacebar to view the commands.
11. Type **interface e0** or **int e0** (or even **int fa0/0**) and press Enter. This will allow you to configure interface Ethernet 0.
12. At the **Router(config-if)#** prompt, type a question mark (?).
13. Type **int s0** (**int s0/0**) or **interface s0** (same as the **interface serial 0** command) and press Enter. This will allow you to configure interface serial 0. Notice that you can go from interface to interface easily.
14. Type **encapsulation ?**.

15. Type **exit**. Notice how this brings you back one level.
16. Press Ctrl+Z. Notice how this brings you out of configuration mode and places you back into privileged mode.
17. Type **disable**. This will put you into user mode.
18. Type **exit**, which will log you out of the router.

Hands-on Lab 4.2: Using the Help and Editing Features

1. Log into the router and go to privileged mode by typing **en** or **enable**.
2. Type a question mark (?).
3. Type **cl?** and then press Enter. Notice that you can see all the commands that start with *cl*.
4. Type **clock ?** and press Enter.



Notice the difference between steps 3 and 4. Step 3 has you type letters with no space and a question mark, which will give you all the commands that start with *cl*. Step 4 has you type a command, space, and question mark. By doing this, you will see the next available parameter.

5. Set the router's clock by typing **clock ?** and, following the help screens, setting the router's time and date.
6. Type **clock ?**.
7. Type **clock set ?**.
8. Type **clock set 10:30:30 ?**.
9. Type **clock set 10:30:30 14 March ?**.
10. Type **clock set 10:30:30 14 March 2002**.
11. Press Enter.
12. Type **show clock** to see the time and date.
13. From privileged mode, type **show access-list 10**. Don't press Enter.
14. Press Ctrl+A. This takes you to the beginning of the line.
15. Press Ctrl+E. This should take you back to the end of the line.
16. Press Ctrl+A, then Ctrl+F. This should move you forward one character.
17. Press Ctrl+B, which will move you back one character.
18. Press Enter, then press Ctrl+P. This will repeat the last command.
19. Press the up arrow key on your keyboard. This will also repeat the last command.
20. Type **sh history**. This shows you the last 10 commands entered.
21. Type **terminal history size ?**. This changes the history entry size. The ? is the number of allowed lines.
22. Type **show terminal** to gather terminal statistics and history size.

23. Type **terminal no editing**. This turns off advanced editing. Repeat steps 14 through 18 to see that the shortcut editing keys have no effect until you type **terminal editing**.
24. Type **terminal editing** and press Enter to re-enable advanced editing.
25. Type **sh run**, then press your Tab key. This will finish typing the command for you.
26. Type **sh start**, then press your Tab key. This will finish typing the command for you.

Hands-on Lab 4.3: Saving a Router Configuration

1. Log into the router and go into privileged mode by typing **en** or **enable**, then press Enter.
2. To see the configuration stored in NVRAM, type **sh start** and press Tab and Enter, or type **show startup-config** and press Enter. However, if no configuration has been saved, you will get an error message.
3. To save a configuration to NVRAM, which is known as startup-config, you can do one of the following:
 - Type **copy run start** and press Enter.
 - Type **copy running**, press Tab, type **start**, press Tab, and press Enter.
 - Type **copy running-config startup-config** and press Enter.
4. Type **sh start**, press Tab, then press Enter.
5. Type **sh run**, press Tab, then press Enter.
6. Type **erase start**, press Tab, then press Enter.
7. Type **sh start**, press Tab, then press Enter. You should get an error message.
8. Type **reload**, then press Enter. Acknowledge the reload by pressing Enter. Wait for the router to reload.
9. Say no to entering setup mode, or just press Ctrl+C.

Hands-on Lab 4.4: Setting Your Passwords

1. Log into the router and go into privileged mode by typing **en** or **enable**.
2. Type **config t** and press Enter.
3. Type **enable ?**.
4. Set your enable secret password by typing **enable secret password** (the third word should be your own personalized password) and then press Enter. Do not add the parameter **password** after the parameter **secret** (this would make your password the word *password*). An example would be **enable secret todd**.
5. Now let's see what happens when you log all the way out of the router and then log in. Log out by pressing Ctrl+Z, and then type **exit** and press Enter. Go to privileged mode. Before you are allowed to enter privileged mode, you will be asked for a password. If you successfully enter the secret password, you can proceed.

6. Remove the secret password. Go to privileged mode, type **config t**, and press Enter. Type **no enable secret** and press Enter. Log out and then log back in again; now you should not be asked for a password.
7. One more password used to enter privileged mode is called the enable password. It is an older, less secure password and is not used if an enable secret password is set. Here is an example of how to set it:

```
config t
enable password todd1
```
8. Notice that the enable secret and enable passwords are different. They cannot be the same.
9. Type **config t** to be at the right level to set your console and auxiliary passwords, then type **line ?**.
10. Notice that the parameters for the line commands are **auxiliary**, **vtty**, and **console**. You will set all three.
11. To set the Telnet or VTY password, type **line vty 0 4** and then press Enter. The 0 4 is the range of the five available virtual lines used to connect with Telnet. If you have an enterprise IOS, the number of lines may vary. Use the question mark to determine the last line number available on your router.
12. The next command is used to set the authentication on or off. Type **login** and press Enter to prompt for a user-mode password when telnetting into the router. You will not be able to telnet into a router if the password is not set.



You can use the **no login** command to disable the user-mode password prompt when using Telnet.

13. One more command you need to set for your VTY password is **password**. Type **password password** to set the password. (*password* is your password.)
14. Here is an example of how to set the VTY password:

```
config t
line vty 0 4
login
password todd
```
15. Set your auxiliary password by first typing **line auxiliary 0** or **line aux 0**.
16. Type **login**.
17. Type **password password**.
18. Set your console password by first typing **line console 0** or **line con 0**.
19. Type **login**.
20. Type **password password**. Here is an example of the last two commands:

```
config t
```

```

line con 0
login
password todd1
line aux 0
login
password todd

```

21. You can add the `Exec-timeout 0 0` command to the `console 0` line. This will stop the console from timing out and logging you out. The command will now look like this:

```

config t
line con 0
login
password todd2
exec-timeout 0 0

```

22. Set the console prompt to not overwrite the command you're typing with console messages by using the command `logging synchronous`.

```

config t
line con 0
logging synchronous

```

Hands-on Lab 4.5: Setting the Hostname, Descriptions, IP Address, and Clock Rate

1. Log into the router and go into privileged mode by typing **en** or **enable**.
2. Set your hostname on your router by using the `hostname` command. Notice that it is one word. Here is an example of setting your hostname:

```

Router#config t
Router(config)#hostname RouterA
RouterA(config)#

```

Notice that the hostname of the router changed as soon as you pressed Enter.

3. Set a banner that the network administrators will see by using the `banner` command.
4. Type **config t**, then **banner ?**.
5. Notice that you can set four different banners. For this lab we are only interested in the login and message of the day (MOTD) banners.
6. Set your MOTD banner, which will be displayed when a console, auxiliary, or Telnet connection is made to the router, by typing

```

config t
banner motd #

```

This is an motd banner

#

7. The preceding example used a # sign as a delimiting character. This tells the router when the message is done. You cannot use the delimiting character in the message itself.

8. You can remove the MOTD banner by typing

config t

no banner motd

9. Set the login banner by typing

config t

banner login #

This is a login banner

#

10. The login banner will display immediately after the MOTD but before the user-mode password prompt. Remember that you set your user-mode passwords by setting the console, auxiliary, and VTY line passwords.

11. You can remove the login banner by typing

config t

no banner login

12. You can add an IP address to an interface with the **ip address** command. You need to get into interface configuration mode first; here is an example of how you do that:

config t

int e0 (you can use **int Ethernet 0** too)

ip address 1.1.1.1 255.255.0.0

no shutdown

Notice that the IP address (1.1.1.1) and subnet mask (255.255.0.0) are configured on one line. The **no shutdown** (or **no shut** for short) command is used to enable the interface. All interfaces are shut down by default.

13. You can add identification to an interface by using the **description** command. This is useful for adding information about the connection. Only administrators see this, not users. Here is an example:

config t

int s0

ip address 1.1.1.2 255.255.0.0

no shut

description Wan link to Miami

14. You can add the bandwidth of a serial link as well as the clock rate when simulating a DCE WAN link. Here is an example:

```

config t
int s0
bandwidth 64
clock rate 64000

```

Hands-on Lab 4.6: Installing SDM on Your Computer

This lab will have you download and install the full SDM program on your computer and then add the demo program inside the program. The links included in this lab were current as of this writing, but they could change at any time.

1. Download and install the latest SDM program from Cisco at the following location: www.cisco.com/pcgi-bin/tablebuild.pl/sdm.
2. After the SDM program is installed on your computer, download the SDM demo at www.cisco.com/pcgi-bin/tablebuild.pl/sdm-tool-demo.
3. Unzip the demo program into a folder of your choice.
4. Copy the `dataFile.zip` file to `C:\`.
5. Make sure you disable your pop-up blockers on your browser.
6. When SDM is installed on a PC, Internet Explorer may display HTML source code when you attempt to launch SDM. To fix this problem, choose **Tools** > **Internet Options** > **Advanced**. Then scroll to the **Security** section, check **Allow Active Content to Run in Files on My Computer**, and click **Apply**. Then relaunch SDM.
7. Click the Cisco SDM icon on your desktop to launch the SDM demo version. Enter the loopback address 127.0.0.1 and click the **Launch** button to start the SDM demo application. You can choose to use HTTPS if you want, and if you do, accept any security certificate notification messages that pop up.
8. The SDM demo will tell you there is debugging running on the router and to disable it because of possible performance degrading, but leave debugging running on the “simulation.” You’re not degrading any router performance.
9. Run through all the tabs: **Configure Interfaces**, **Routing Protocols**, **Create a DHCP Pool**, and so on—everything you can find. Spend some time getting to know SDM.
10. Understand that not all features are supported on the demo, but it is better than not having SDM and the demo actually works pretty well.
11. If you want to practice using certificates, use the `ca.cer` and `router.cer` certificates to demonstrate importing certificate authority (CA) and router certificates. You could get these certificates from `SDM_demo_tool.zip`.



The SDM files on Cisco.com change constantly, which changes the lab written above. Please see my forum and blog at www.lammle.com for the up-to-minute information on SDM and how to install and run the demo.